
LONGITUDINAL EMPLOYER - HOUSEHOLD DYNAMICS

INTERNAL DOCUMENT NO. IP-LEHD-ECF-3.1.49

LEHD-ECF Technical Documentation
(Nonconfidential)

Code version : 3.1.49
Author : Kevin McKinney and Lars Vilhuber
Date : March 11, 2005

Contents

- 1 ECF 2**
- 1.1 Overview 6
- 1.1.1 General Overview 6
- 1.1.2 Input Files 6
- 1.1.3 Program Overview 7
- 1.1.4 SAS Program legacy name mapping 7
- 1.1.5 Files Created 7
- 1.1.6 Details on intermediate files 9
- 1.1.7 Variables Created 13
- 1.2 NAICS codes on the ECF 20
- 1.2.1 LDB versus LEHD NAICS backcoding 20
- 1.2.2 Variable List 21
- 1.2.3 Coding of MISS and SRC 22
- 1.2.4 NAICS algorithm precedence ordering 23
- 1.2.5 ESO and FNL variables 24
- 1.2.6 Employment Flag Variable Codes 24
- 1.2.7 Multi-Unit Code or MEEI 25
- 1.2.8 Auxiliary Code 26
- 1.3 History and directions for the future 26
- 1.3.1 History 26
- 1.3.2 Directions for the Future 27
- 1.4 How to run diagnostics 27
- 1.4.1 When to run the diagnostics 27
- 1.4.2 Modifying the parameters file 27
- 1.4.3 Which file to run 27
- 1.5 Details 28
- 1.5.1 Control programs 28
- 1.5.2 SAS programs used 43
- 1.5.3 SAS macros used 204
- 1.5.4 SAS layouts used 270
- 1.5.5 SAS formats used 271
- 1.5.6 Scripts used 272
- 1.5.7 Total code lines for this process 273

Chapter 1

ECF

Contents

1.1 Overview	6
1.1.1 General Overview	6
1.1.2 Input Files	6
1.1.3 Program Overview	7
1.1.4 SAS Program legacy name mapping	7
1.1.5 Files Created	7
1.1.5.1 ecf_ST_seinunit.sas7bdat	7
1.1.5.2 ecf_ST_sein.sas7bdat	9
1.1.5.3 ecf_ST_fuzz_sein.sas7bdat	9
1.1.5.4 ecf_ST_fuzz_seinunit.sas7bdat	9
1.1.5.5 ecf_ST_leg.sas7bdat	9
1.1.6 Details on intermediate files	9
1.1.6.1 ecf_stacked_01.sas7bdat	9
1.1.6.2 count_data_02.sas7bdat	9
1.1.6.3 no_master_03.sas7bdat	9
1.1.6.4 sein_totals_04.sas7bdat	10
1.1.6.5 sein_list_ui_04.sas7bdat	10
1.1.6.6 sein_list_202_04.sas7bdat	10
1.1.6.7 seinunit_202_UI_04.sas7bdat	10
1.1.6.8 best_vars_05.sas7bdat	10
1.1.6.9 special_handle_list_06.sas7bdat	10
1.1.6.10 alldata_06.sas7bdat	10
1.1.6.11 special_handle_history_06.sas7bdat	10
1.1.6.12 special_handle_07.sas7bdat	10
1.1.6.13 ST_employer_char_unit.sas7bdat	10
1.1.6.14 ST_employer_char_unit2.sas7bdat	10
1.1.6.15 seinunit_county_09.sas7bdat	10
1.1.6.16 seinunit_msapmsa_09.sas7bdat	11
1.1.6.17 seinunit_state_09.sas7bdat	11
1.1.6.18 seinunit_subctygeo_09.sas7bdat	11
1.1.6.19 seinunit_wib_09.sas7bdat	11
1.1.6.20 seinunit_wide_09.sas7bdat	11
1.1.6.21 seinunit_long_09.sas7bdat	11
1.1.6.22 seinunit_county_10.sas7bdat	11
1.1.6.23 seinunit_ein_10.sas7bdat	11
1.1.6.24 seinunit_naics_10.sas7bdat	11
1.1.6.25 seinunit_owner_code_10.sas7bdat	11
1.1.6.26 seinunit_sic_10.sas7bdat	11

1.1.6.27	seinunit_long_10.sas7bdat	12
1.1.6.28	sein_modes_11.sas7bdat	12
1.1.6.29	sein_wide_12.sas7bdat	12
1.1.6.30	sein_long_12.sas7bdat	12
1.1.6.31	sein_missing_13.sas7bdat	12
1.1.6.32	sein_mode_fill_13.sas7bdat	12
1.1.6.33	sein_missing_14.sas7bdat	12
1.1.6.34	sein_long_14.sas7bdat	12
1.1.6.35	sicfreq_14.sas7bdat	12
1.1.6.36	sicfreq2_14.sas7bdat	12
1.1.6.37	seinunit_long_15.sas7bdat	12
1.1.6.38	il_employer_char_unit3.sas7bdat	12
1.1.6.39	weights_sein_17.sas7bdat	12
1.1.6.40	weight_totals_ss_18.sas7bdat	13
1.1.6.41	weights_sein_18.sas7bdat	13
1.1.6.42	il_employer_char_unit4.sas7bdat	13
1.1.6.43	new_sein_fuzz_19.sas7bdat	13
1.1.6.44	new_seinunit_fuzz_19.sas7bdat	13
1.1.7	Variables Created	13
1.1.7.1	01_read_all.sas	13
1.1.7.2	02_num_records.sas	14
1.1.7.3	03_rm_master.sas	14
1.1.7.4	04_sein_totals.sas	15
1.1.7.5	05_best_vars.sas	15
1.1.7.6	06_select_records.sas	16
1.1.7.7	07_special_handle.sas	17
1.1.7.8	08_distribute.sas	17
1.1.7.9	09_seinunit_wide.sas	17
1.1.7.10	10_fill_sync.sas	18
1.1.7.11	11_sein_mode_calc.sas	18
1.1.7.12	12_sein_wide.sas	18
1.1.7.13	13_sein_fill.sas	19
1.1.7.14	14_impute_sein_industry.sas	19
1.1.7.15	15_impute_seinunit_industry.sas	19
1.1.7.16	16_seinunit_yq_chars.sas	19
1.1.7.17	17_weight_calc_01.sas	19
1.1.7.18	18_weight_calc_02.sas	19
1.1.7.19	19_seinunit_fuzz.sas	20
1.1.7.20	20_sein_file.sas	20
1.1.7.21	21_fuzz_update.sas	20
1.2	NAICS codes on the ECF	20
1.2.1	LDB versus LEHD NAICS backcoding	20
1.2.2	Variable List	21
1.2.3	Coding of MISS and SRC	22
1.2.3.1	MISS Variable Codes	22
1.2.3.2	SRC Variable Codes	23
1.2.4	NAICS algorithm precedence ordering	23
1.2.5	ESO and FNL variables	24
1.2.6	Employment Flag Variable Codes	24
1.2.7	Multi-Unit Code or MEEI	25
1.2.8	Auxiliary Code	26
1.3	History and directions for the future	26
1.3.1	History	26
1.3.2	Directions for the Future	27
1.4	How to run diagnostics	27
1.4.1	When to run the diagnostics	27

1.4.2	Modifying the parameters file	27
1.4.3	Which file to run	27
1.5	Details	28
1.5.1	Control programs	28
1.5.1.1	ctrlprogs/control_ecf.sas	28
1.5.1.2	ctrlprogs/parameters_ecf.sas	40
1.5.2	SAS programs used	43
1.5.2.1	library/sasprogs/01_ecf.sas	43
1.5.2.2	library/sasprogs/02_ecf.sas	45
1.5.2.3	library/sasprogs/03_ecf.sas	49
1.5.2.4	library/sasprogs/04_ecf.sas	54
1.5.2.5	library/sasprogs/05_ecf.sas	57
1.5.2.6	library/sasprogs/06_ecf.sas	58
1.5.2.7	library/sasprogs/07_ecf.sas	61
1.5.2.8	library/sasprogs/08_ecf.sas	66
1.5.2.9	library/sasprogs/09_ecf.sas	72
1.5.2.10	library/sasprogs/10_ecf.sas	75
1.5.2.11	library/sasprogs/11_ecf.sas	80
1.5.2.12	library/sasprogs/12_ecf.sas	83
1.5.2.13	library/sasprogs/13_ecf.sas	96
1.5.2.14	library/sasprogs/14_ecf.sas	98
1.5.2.15	library/sasprogs/15_ecf.sas	108
1.5.2.16	library/sasprogs/16_ecf.sas	116
1.5.2.17	library/sasprogs/17_ecf.sas	118
1.5.2.18	library/sasprogs/18_ecf.sas	122
1.5.2.19	library/sasprogs/19_ecf.sas	126
1.5.2.20	library/sasprogs/20_ecf.sas	130
1.5.2.21	library/sasprogs/21_ecf.sas	138
1.5.2.22	library/sasprogs/22_ecf.sas	141
1.5.2.23	library/sasprogs/23_ecf.sas	147
1.5.2.24	library/sasprogs/24_ecf.sas	157
1.5.2.25	library/sasprogs/25_ecf.sas	161
1.5.2.26	library/sasprogs/26_ecf.sas	163
1.5.2.27	library/sasprogs/27_ecf.sas	172
1.5.2.28	library/sasprogs/28_ecf.sas	174
1.5.2.29	library/sasprogs/29_ecf.sas	177
1.5.2.30	library/sasprogs/30_ecf.sas	180
1.5.2.31	library/sasprogs/31_ecf.sas	182
1.5.2.32	library/sasprogs/01_ecfqa.sas	184
1.5.2.33	library/sasprogs/02_ecfqa.sas	185
1.5.2.34	library/sasprogs/03_ecfqa.sas	186
1.5.2.35	library/sasprogs/04_ecfqa.sas	192
1.5.2.36	library/sasprogs/05_ecfqa.sas	198
1.5.2.37	library/sasprogs/06_ecfqa.sas	199
1.5.2.38	library/sasprogs/07_ecfqa.sas	200
1.5.2.39	library/sasprogs/08_ecfqa.sas	201
1.5.2.40	library/sasprogs/test.sas	203
1.5.3	SAS macros used	204
1.5.3.1	library/macros/addnoise.sas	204
1.5.3.2	library/macros/bldstr.sas	206
1.5.3.3	library/macros/check_ein.sas	207
1.5.3.4	library/macros/choose_gal2.sas	210
1.5.3.5	library/macros/choose_gal.sas	212
1.5.3.6	library/macros/create_esgal.sas	213
1.5.3.7	library/macros/create_pseudo_esgal.sas	215
1.5.3.8	library/macros/ecfdiag_01.sas	216

1.5.3.9	library/macros/ecfdiag_02.sas	219
1.5.3.10	library/macros/ecfdiag_03.sas	219
1.5.3.11	library/macros/ecfdiag_04.sas	220
1.5.3.12	library/macros/ecfdiag_05.sas	221
1.5.3.13	library/macros/ecfdiag_06.sas	222
1.5.3.14	library/macros/ecfdiag_07.sas	223
1.5.3.15	library/macros/ecfdiag_08.sas	224
1.5.3.16	library/macros/ecfdiag_09.sas	226
1.5.3.17	library/macros/ecfdiag_10.sas	228
1.5.3.18	library/macros/ecfdiag_11.sas	229
1.5.3.19	library/macros/ecfdiag_12.sas	230
1.5.3.20	library/macros/ecfdiag_13.sas	232
1.5.3.21	library/macros/ecfdiag_14.sas	233
1.5.3.22	library/macros/ecfdiag_15.sas	233
1.5.3.23	library/macros/ecfdiag_16.sas	234
1.5.3.24	library/macros/ecfdiag_17.sas	234
1.5.3.25	library/macros/ecfdiag_18.sas	235
1.5.3.26	library/macros/ecfdiag_19.sas	236
1.5.3.27	library/macros/ecfdiag_20.sas	237
1.5.3.28	library/macros/ecfdiag_21.sas	239
1.5.3.29	library/macros/ecfdiag_22.sas	240
1.5.3.30	library/macros/ecfdiag_23.sas	241
1.5.3.31	library/macros/ecfdiag_24.sas	242
1.5.3.32	library/macros/ecfdiag_25.sas	244
1.5.3.33	library/macros/ecfdiag_26.sas	245
1.5.3.34	library/macros/ecfdiag_27.sas	249
1.5.3.35	library/macros/ecfdiag_28.sas	249
1.5.3.36	library/macros/ecfdiag_29.sas	251
1.5.3.37	library/macros/ecfdiag_30.sas	252
1.5.3.38	library/macros/ecfdiag_31.sas	253
1.5.3.39	library/macros/ecfqa_checkfuzz.sas	254
1.5.3.40	library/macros/fixfips.sas	256
1.5.3.41	library/macros/fzexist.sas	256
1.5.3.42	library/macros/indlkup2.sas	257
1.5.3.43	library/macros/indlkup.sas	260
1.5.3.44	library/macros/modevar.sas	261
1.5.3.45	library/macros/qloop.sas	262
1.5.3.46	library/macros/qtime.sas	262
1.5.3.47	library/macros/set_es.sas	263
1.5.3.48	library/macros/set_ldb_qa.sas	263
1.5.3.49	library/macros/set_ldb.sas	265
1.5.3.50	library/macros/state_specific_cleanup.sas	266
1.5.3.51	library/macros/state_specific_keeprec.sas	267
1.5.3.52	library/macros/upnum.sas	268
1.5.3.53	library/macros/yqreal.sas	269
1.5.3.54	library/macros/yrqtr.sas	269
1.5.4	SAS layouts used	270
1.5.5	SAS formats used	271
1.5.6	Scripts used	272
1.5.7	Total code lines for this process	273

1.1 Overview

1.1.1 General Overview

The Employer Characteristics File (ECF) consolidates most firm level information (size, location, industry, etc.) into two easily accessible files. The firm or SEIN level file contains one record for every YEAR QUARTER a firm is present in either the ES-202 or the UI, with more detailed information available for the establishments of multi-unit firms in the SEIN SEINUNIT file. The SEIN file is built up from the SEINUNIT file and contains no additional information, but should be viewed merely as an easier and/or more efficient way to access firm level data.

SEIN level file unique record identifier: SEIN YEAR QUARTER

SEIN level file sort order: SEIN YEAR QUARTER

SEIN level file indexes: SEIN, SEIN_YEAR.QUARTER

SEINUNIT level file unique record identifier: SEIN SEINUNIT YEAR QUARTER

SEINUNIT level file sort order: SEIN SEINUNIT YEAR QUARTER

SEINUNIT level file indexes: SEIN, SEIN_SEINUNIT_YEAR.QUARTER,
SEIN_YEAR.QUARTER, SEIN_YEAR.QUARTER_SEINUNIT

1.1.2 Input Files

- The ES202 data from the states is the primary input to the ECF file creation process.
- UI data is also used to supplement information on the ES202. As part of the creation of the Employment History File (EHF), `ehf_sein_employment` is created. This file contains E (end of period employment), B (beginning of period employment), M (employed anytime in the quarter), and W1 (total wages) calculated similarly to the same measures on the QWI.
- GAL data containing lat/long coordinates of the establishments, plus county, wib and pmsa geo also.
- Existing fuzz files must be available if data for the state has been officially released.
- SIC and NAICS impute datasets:
 1. 3 digit (`parms_us_imp_sic_sic3`, formerly called `sic3_impute`) and 2 digit (`parms_us_imp_sic_sic2`, formerly called `sic2_impute.sas7bdat`) SIC impute datasets are used in the first modules,
 2. 6 digit NAICS to 4 digit SIC probabilistic crosswalk (`parms_us_imp_ncs1997_sic`, formerly called `naics_sic_9`, and `parms_us_imp_sic_ncs1997`) is used after the merge to the GAL.
 3. NAICS 2002 to 1997 crosswalks are used (`parms_naics_imp_ncs1997_ncs2002` and `parms_naics_imp_ncs2002_ncs1997`).
- Various formats are used:
 1. multiformats for SIC and NAICS are used from the format library,
 2. SIC division formats,
 3. EIN checking routine,
 4. county formats that vary from state to state.

Format files are all located in `/programs/projects/auxiliary/Formats`.

- BLS-derived control totals, produced by the EHF.

1.1.3 Program Overview

First data is read in from the yearly ES202 files and stacked one on top of the other. General and state specific consistency checks are then performed. The COUNTY, NAICS, and EIN data are checked for invalid values. The SIC invalid check is a little more sophisticated. If a 4 digit SIC code is present, but is not valid, then the SIC code undergoes a conditional impute based on the first 2 or 3 digits. If the first 2 or 3 digits are not valid either, then SIC is set to missing (this value will eventually be filled).

The ES202 data contains a “master” record for multi-unit firms that must be removed. Information in the master record is preserved if data is not available in the establishment records (I initially allocate the data equally to each establishment). Various inconsistencies in the record structure are also dealt with, such as 2 records (master and establishment) appearing for a single-unit.

The UI data is integrated with the ES202 data and totals are calculated at the SEIN YEAR QUARTER level.

Using both UI and ES202 data I create a “best” series of variables for payroll and employment.

The allocation process implemented above (master to establishments) does not incorporate any information on the structure of the firm. A flat prior is used in the allocation process (each establishment is assumed to have equal employment and payroll). I improve on this by examining firms with allocated data that previously reported as a multi-unit. I then use the structure of their reports from a previous quarter to allocate payroll and employment. The new records are integrated back into the data, hopefully improving longitudinal consistency at the establishment level.

At this point, the SEIN YEAR QUARTER SEINUNIT dataset record structure is finalized.

The GAL is brought into the ECF (this used to be the separate LEG process).

The COUNTY, SIC, NAICS, and EIN data are transformed from long to wide format for each SEINUNIT. I use this dataset to fill missing values in these variables with information from other periods for the same establishment.

The modal COUNTY, SIC, NAICS, OWNER_CODE, and EIN are calculated (both establishment and employment weighted) for each SEIN in a given YEAR and QUARTER.

The SEIN level mode variables (SIC, NAICS, etc) are then transformed from long to wide and the missing values are filled with data from the closest YEAR and QUARTER, if available.

At this point, if an SEIN mode variable has a missing value, then that missing value must be present for every YEAR and QUARTER. The distribution of employment across 4 digit SIC in 1997 is calculated and is used to impute the industry code for each SEIN with missing SIC. These SIC codes are also assigned to the SEINUNIT level data.

The weights are calculated, based on the expanded BLS controltotals acquired from the EHF.

The final step is to apply fuzz factors to each dataset. The fuzz factor process is done separately for the SEIN and the SEINUNIT data. Once this is completed the datasets are written to their final location and the master fuzz files are updated.

1.1.4 SAS Program legacy name mapping

Please note that program names have been changed since legacy version 3.1.0. Table 1.1 lists a correspondence of the old descriptive names to the numeric names in the production system. The old descriptive names are also coded in the first line of every program.

1.1.5 Files Created

Most temporary SAS datasets have the number of the program that created them in the name. Some final output datasets were renamed, see Table 1.2.

1.1.5.1 ecf_ST_seinunit.sas7bdat

Created by 19_seinunit_fuzz.sas

The final version of the SEIN SEINUNIT YEAR QUARTER ECF.

Table 1.1: Name conversion between legacy programs, NAICS-adapted, and current programs

Legacy name (3.1.0)		NAICS (3.1.08-kev)		3.1.15+
01_read_all	→	01_read_es202	→	01_ecf
	NEW	02_naics_clean	→	02_ecf
	NEW	03_naics_aux_clean	→	03_ecf
	NEW	04_sic_clean	→	04_ecf
	NEW	05_merge_ind	→	05_ecf
02_num_records	→	06_num_records	→	06_ecf
03_rm_master	→	07_rm_master	→	07_ecf
04_sein_totals	→	08_sein_totals	→	08_ecf
05_best_vars	→	09_best_vars	→	09_ecf
06_select_records	→	10_select_records	→	10_ecf
07_special_handle	→	11_special_handle	→	11_ecf
08_distribute	→	12_distribute	→	12_ecf
01_leg	→	13_leg1	→	13_ecf
02_leg	→	14_leg2	→	14_ecf
03_leg	→	16_leg3	→	15_ecf
04_leg	→	16_leg4	→	16_ecf
	NEW	17_naics_ldb_clean	→	17_ecf
09_seinunit_wide	→	18_seinunit_wide	→	18_ecf
10_fill_sync	→	19_seinunit_fill	→	19_ecf
	SPLIT	20_seinunit_sync	→	20_ecf
11_sein_mode_calc	→	21_sein_mode_calc	→	21_ecf
12_sein_wide	→	22_sein_wide	→	22_ecf
13_sein_fill	→	23_sein_fill	→	23_ecf
14_impute_sein_industry	→	24_impute_sein_industry	→	24_ecf
15_impute_seinunit_industry	→	25_impute_seinunit_industry	→	25_ecf
16_seinunit_yq_chars	→	26_seinunit_yq_chars	→	26_ecf
17_weight_calc_01	→	27_weight_calc_01	→	27_ecf
18_weight_calc_02	→	28_weight_calc_02	→	28_ecf
19_seinunit_fuzz	→	29_seinunit_fuzz	→	29_ecf
20_sein_file	→	30_sein_file	→	30_ecf
21_fuzz_update	→	31_fuzz_update	→	31_ecf
01_legqa	→			01_ecfqa
02_legqa	→			02_ecfqa
03_legqa	→			03_ecfqa
04_legqa	→			04_ecfqa

Table 1.2: Changed dataset names

sic3_impute	→	parms.us_sic3impute
	→	parms.us_imp_sic_sic3
sic2_impute	→	parms.us_sic2impute
	→	parms.us_imp_sic_sic2
naics_sic_9	→	parms.us_naics_sic
	→	parms.us_imp_ncs1997_sic
INPUTS.&state._master_sein_fuzz	→	INPUTS.ecf.&state._fuzz_sein
INPUTS.&state._master_seinunit_fuzz	→	INPUTS.ecf.&state._fuzz_seinunit
OUTPUTS.&state._master_sein_fuzz	→	OUTPUTS.ecf.&state._fuzz_sein
OUTPUTS.&state._master_seinunit_fuzz	→	OUTPUTS.ecf.&state._fuzz_seinunit
OUTPUTS.ecf_seinunit.&state.	→	OUTPUTS.ecf.&state._seinunit
OUTPUTS.ecf_sein.&state.	→	OUTPUTS.ecf.&state._sein
&state._leg.structure	→	OUTPUTS.ecf.&state._leg.structure
	→	eliminated (3.1.12)
leg.&state.	→	OUTPUTS.ecf.&state._leg
xwlc_e&year	→	gal.&state._xwlc_e&year
gal_us_subcty	→	parms.us_wib_subcty

1.1.5.2 ecf_ST_sein.sas7bdat

Created by 20_sein_file.sas

The final version of the SEIN YEAR QUARTER ECF.

1.1.5.3 ecf_ST_fuzz_sein.sas7bdat

Created by 21_fuzz_update.sas

If the SEIN file does not exist it is created, otherwise the file is updated through a merge with new_sein_fuzz_19.sas7bdat.

1.1.5.4 ecf_ST_fuzz_seinunit.sas7bdat

Created by 21_fuzz_update.sas

If the SEIN SEINUNIT file does not exist it is created, otherwise the file is updated through a merge with new_seinunit_fuzz_19.sas7bdat.

1.1.5.5 ecf_ST_leg.sas7bdat

The old LEG. Useful for research.

1.1.6 Details on intermediate files

1.1.6.1 ecf_stacked_01.sas7bdat

Stacked data from the ES202. Contains master and subunits

1.1.6.2 count_data_02.sas7bdat

adds number of record counts and data availability flags to ecf_stacked_01.sas7bdat

1.1.6.3 no_master_03.sas7bdat

Removes the master record. File is now in the form SEIN YEAR QUARTER SEINUIT

1.1.6.4 sein_totals_04.sas7bdat

SEIN YEAR QUARTER totals from the ES202

1.1.6.5 sein_list_ui_04.sas7bdat

A list of the SEIN's that ever appear on the UI

1.1.6.6 sein_list_202_04.sas7bdat

A list of the SEIN's that ever appear on the ES202

1.1.6.7 seinunit_202_UI_04.sas7bdat

SEIN YEAR QUARTER SEINUNIT file containing payroll and employment totals from both the UI and ES202. If an observation is only from the UI then there is no SEINUNIT information.

1.1.6.8 best_vars_05.sas7bdat

Improved payroll and employment measures at the SEIN YEAR QUARTER SEINUNIT level.

1.1.6.9 special_handle_list_06.sas7bdat

SEIN YEAR QUARTER records that will have SEINUNIT record structure imputed

1.1.6.10 alldata_06.sas7bdat

A copy of best_vars_05.sas7bdat with several new variables added.

1.1.6.11 special_handle_history_06.sas7bdat

SEIN YEAR QUARTER SEINUNIT dataset for each SEIN that has a record in special_handle_list_06.sas7bdat. The complete history of the SEIN is included in this dataset.

1.1.6.12 special_handle_07.sas7bdat

The new record structure created for records in special_handle_list_06.sas7bdat by using information from another quarter. The number of SEIN YEAR QUARTER records is unchanged.

1.1.6.13 ST_employer_char_unit.sas7bdat

Created by 08.distribute.sas

The structure of the ECF is set when this file is created. The employment and payroll variables have been completed.

1.1.6.14 ST_employer_char_unit2.sas7bdat

Created by 09_seinunit_wide.sas

The geography variables have been brought in from the LEG.

1.1.6.15 seinunit_county_09.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with LEG county and employment variables. Used to calculate the mode.

1.1.6.16 seinunit_msapmsa_09.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with LEG msapmsa and employment variables. Used to calculate the mode.

1.1.6.17 seinunit_state_09.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with LEG state and employment variables. Used to calculate the mode.

1.1.6.18 seinunit_subctygeo_09.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with LEG subctygeo and employment variables. Used to calculate the mode.

1.1.6.19 seinunit_wib_09.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with LEG wib codes and employment variables. Used to calculate the mode.

1.1.6.20 seinunit_wide_09.sas7bdat

SEIN SEINUNIT file with SIC, NAICS, county, ownership code, and EIN values placed in arrays. Used to fill missing values.

1.1.6.21 seinunit_long_09.sas7bdat

SEIN SEINUNIT YEAR QUARTER file containing the general structure of the data and employment variables. Used in program 10 to rebuild the structure from the wide file.

1.1.6.22 seinunit_county_10.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with county codes and employment variables. Used to calculate the mode.

1.1.6.23 seinunit_ein_10.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with the EIN and employment variables. Used to calculate the mode.

1.1.6.24 seinunit_naics_10.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with NAICS codes and employment variables. Used to calculate the mode.

1.1.6.25 seinunit_owner_code_10.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with ownership codes and employment variables. Used to calculate the mode.

1.1.6.26 seinunit_sic_10.sas7bdat

SEIN SEINUNIT YEAR QUARTER file with SIC codes and employment variables. Used to calculate the mode.

1.1.6.27 seinunit_long_10.sas7bdat

SEIN SEINUNIT YEAR QUARTER file containing the missing filled data for SIC, NAICS, county, ownership code, and EIN.

1.1.6.28 sein_modes_11.sas7bdat

SEIN YEAR QUARTER file containing the modal values for SIC, NAICS, county, ownership code, EIN, wib, msapmsa, LEG state, LEG county, and subctygeo.

1.1.6.29 sein_wide_12.sas7bdat

SEIN file containing the mode variable values in arrays.

1.1.6.30 sein_long_12.sas7bdat

SEIN YEAR QUARTER file containing the structure of the data.

1.1.6.31 sein_missing_13.sas7bdat

File containing the list of SEIN's with missing values for SIC.

1.1.6.32 sein_mode_fill_13.sas7bdat

SEIN YEAR QUARTER file containing the mode variables with filled missing values.

1.1.6.33 sein_missing_14.sas7bdat

Same record count as sein_missing_13.sas7bdat
SEIN file containing imputed SIC.

1.1.6.34 sein_long_14.sas7bdat

SEIN YEAR QUARTER file containing missing filled mode variables.

1.1.6.35 sicfreq_14.sas7bdat

File containing the employment in each 4 digit SIC

1.1.6.36 sicfreq2_14.sas7bdat

Slightly transformed version of sicfreq_14.sas7bdat. Contains the CDF of employment by SIC.

1.1.6.37 seinunit_long_15.sas7bdat

SEIN YEAR QUARTER SEINUNIT file containing the missing value cleaned SIC, NAICS, county, ownership code, and EIN.

1.1.6.38 il_employer_char_unit3.sas7bdat

Created by 16_seinunit_yq_chars.sas
SEIN YEAR QUARTER SEINUNIT file containing all variables.

1.1.6.39 weights_sein_17.sas7bdat

YEAR QUARTER SEIN file containing the results of the first stage of the weights process.

1.1.6.40 **weight_totals_ss_18.sas7bdat**

YEAR QUARTER summary file containing various totals necessary for the weights calculation.

1.1.6.41 **weights_sein_18.sas7bdat**

SEIN YEAR QUARTER file containing the final weights

1.1.6.42 **il_employer_char_unit4.sas7bdat**

Created by 18_weights_calc_02.sas

SEIN SEINUNIT YEAR QUARTER file containing all variables plus the new weights.

1.1.6.43 **new_sein_fuzz_19.sas7bdat**

SEIN file containing the new fuzz factors

1.1.6.44 **new_seinunit_fuzz_19.sas7bdat**

SEIN SEINUNIT file containing the new fuzz factors

1.1.7 **Variables Created**

Please note that program names in this section refer to the programs as present in the legacy version 3.1.0. Table 1.1 lists a correspondence of the old descriptive names to the numeric names in the production system.

NOTE to LARS: this should be replaced/augmented by PROC CONTENTS of the actual files. See the data codebook generating programs.

1.1.7.1 **01_read_all.sas**

sein Variables read in from the ES202 yearly files.

12 digit firm identifier (first 2 digits are the state FIPS code)

year

quarter

seinunit 5 digit code identifying the establishment. Generally used in combination with the SEIN to uniquely identify an establishment. The identifier itself is only unique within a firm or SEIN.

owner_code see ES_OWNER_CODE

EIN

county

SIC

NAICS

empl_month1

empl_month2

empl_month3

total_wages End of variables read in from the ES202 yearly files.

Sein_bad 0 = SEIN contains only characters 0-9
1 = SEIN contains a character outside the above range

Ein_bad 0 = EIN contains only characters 0-9
1 = EIN contains a character outside the above range

Valid_ein 0 = first 2 digits of EIN do not represent a valid IRS Revenue district code
1 = first 2 digits are valid

Ein_defect 0 = no defect found
1 = EIN it is all nines or all zeros
2 = ein_bad=1, EIN contains characters outside the range 0-9
3 = EIN is a 7 digit or less number. An EIN must be at least eight characters
4 = valid_ein=0, the first two digits of the EIN do not represent a valid IRS Revenue district code

Sic_invalid 0 = SIC is OK
1 = SIC not valid
2 = first 2 digits valid, last 2 digits imputed
3 = first 3 digits valid, last digit imputed

1.1.7.2 02_num_records.sas

NUM.RECORDS 1-N = the number of records for each SEIN in a given year and quarter

All_miss_(pay,emp1,emp2,emp3,sic,county) 0 = at least one or more subunits has data
1 = all subunits have missing data

1.1.7.3 03_rm_master.sas

num_estabs 1-N = the number of establishments for each SEIN in a given year and quarter

multi_unit 0 = not a multi unit
1 = multi unit

impute_(wage,emp1,emp2,emp3,sic,county) 0 = data not available or imputation unnecessary
1 = data available in master record and no data in subunits

no_(wages,emp1,emp2,emp3,sic,county) 0 = data available in either master record or subunits
1 = no data in either master record or subunits

master_(wage,emp1,emp2,emp3,sic,county) Information contained in the master record is stored here

seinunit_type 0 = seinunit="00000"
1 = seinunit~="00000"

1.1.7.4 04_sein_totals.sas

seinsize_m variables read in from the UI SEIN YEAR QUARTER summary file.

Count of PIK level wage records that appear at an SEIN in a given YEAR QUARTER.

seinsize_b Count of PIK level wage records that appear at an SEIN in both the current and previous YEAR QUARTER.

seinsize_e Count of PIK level wage records that appear at an SEIN in both the current and subsequent YEAR QUARTER.

Payroll Sum of earnings for PIK level wage records at the SEIN in a given YEAR QUARTER.

ever_(multi,wages,emp1,emp2,emp3) 0 = the SEIN never reports data on the ES202

1 = the SEIN is a multi unit at some time or reports payroll or employment at some time during the observed period on the ES202.

sein_(emp1,emp2,emp3,wages) SEIN level totals for payroll and employment from the ES202

multi_first_year The first year when an SEIN appears as a multi unit on the ES202

multi_first_quarter The first quarter when an SEIN appears as a multi unit on the ES202

in_UI 0 = SEIN is not on the UI in a given year and quarter

1 = SEIN appears on the UI in given year and quarter

in_202 0 = SEIN is not on the ES202 a given year and quarter

1 = SEIN appears on the ES202 in a given year and quarter

source 1 = UI only

2 = ES202 only

3 = both UI and ES202

ever_202 0 = not on ES202

1 = SEIN appears on the ES202 at some time during observed period

yr_qtr A 6 character sequential year variable. Format is YYYY:Q. A 4 digit year, a colon, and a 1 digit quarter.

1.1.7.5 05_best_vars.sas

emp(1,2,3)_UI Attempt to create the best possible approximation of ES202 employment and payroll using UI data.

Emp1_UI = seinsize_b if available, then seinsize_e, and finally seinsize_m.

Emp2_UI = seinsize_b if available, then seinsize_e, and finally seinsize_m.

Emp1_UI = seinsize_e if available, then seinsize_b, and finally seinsize_m.

best_(wages,emp1,emp2,emp3) My best estimate of payroll and employment for a subunit using as much information available in the UI and ES202. I use both contemporaneous information and information about the firm in other years and quarters. If information is available in the ES202 then that data takes precedence over information in the UI.

best_flag NOTE: The best_flag variable when combined with the structure_fix variable can be used to identify the type of edits and data source of the best_xx variables.

- 0 = no wage or employment information on the ES202 or UI
- 1 = SU, ES202 wages, but ES202 employment is zero
- 2 = SU, ES202 wages, but ES202 employment is missing
- 3 = SU, no ES202 wages, but ES202 employment is available
- 4 = SU, ES202 wages and employment is greater than zero
- 5 = SU, no ES202 wages or employment, UI available
- 6 = SU, not in ES202 and UI available
- 7 = MU, ES202 wages, but ES202 employment is zero
- 8 = MU, ES202 wages, but ES202 employment is missing
- 9 = MU, no ES202 wages, but ES202 employment is available
- 10 = MU, ES202 wages and employment is greater than zero
- 11 = MU, no ES202 wages or employment, UI available

info_202 0 = no wages and employment on ES202

- 1 = wages and no employment on ES202
- 2 = wages and no employment on ES202
- 3 = wages and employment on ES202

noemp_202 0 = positive ES202 employment

- 1 = employment is not >0 on the ES202

emp_202_miss 0 = not in the ES202 and non-missing ES202 employment

- 1 = in the ES202 and all ES202 employment is missing.

1.1.7.6 06_select_records.sas

special_handle 0 = no special handling required

- 1 = in_UI=1 and in_202=0 and ever_multi=1
- 2 = in_UI=0 and in_202=1 and impute_data=1
- 3 = in_UI=1 and in_202=1 and no_data=1 and multi_unit=1
- 4 = in_UI=1 and in_202=1 and impute_data=1

no_get_data 0 = get_XX=1 for at least one variable

- 1 = get_XX=0 for all variables

data_avail 0 = no data available

- 1 = in_202=1 and some subunit data available that period

impute_data 0 = no allocation of master to subunit that period

- 1 = allocation of master to subunit that period

no_data 0 = data available

- 1 = no data in master or subunit available that period

get_(wages,emp1,emp2,emp3) 0 = special_handle=0 or special_handle=1 and no subunit wages available in other periods

- 1 = special_handle<0 and subunit data is available in other periods

(wages,emp1,emp2,emp3)_202 Renamed sein_XX variables on the special_handle.06.sas7bdat dataset. This is necessary in the next program when I match a record with missing subunit information the to another record in another year and quarter.

Wages_UI Payroll is renamed similarly to emp(1,2,3)_UI variables.

1.1.7.7 07_special_handle.sas

qtime_master Continuous quarter time from 1985 quarter 1 for the record for which I am trying to determine subunit structure.

qtime_first The first quarter in continuous time that an SEIN appears as a multi unit

year_found The closest year that contains subunit structure

quarter_found The closest quarter that contains subunit structure

Stop 0 = record not found

1 = record with subunit structure found

1.1.7.8 08_distribute.sas

best_(wages,emp1,emp2,emp3) Update of original values computed in 05_best_vars.sas. My best estimate of payroll and employment for a subunit using as much information available in the UI and ES202. I use both contemporaneous information and information about the firm in other years and quarters. If information is available in the ES202 then that data takes precedence over information in the UI.

sein_best_(wages, emp1, emp2, emp3) SEIN YEAR QUARTER summaries of the best_XX variables.

structure_fix NOTE: The best_flag variable when combined with the structure_fix variable can be used to identify the type of edits and data source of the best_xx variables.

0 = record not selected for structure imputation

1 = record selected for structure imputation

1.1.7.9 09_seinunit_wide.sas

leg_state See the LEG documentation for more information on these variables

leg_county

leg_wib

leg_msapmsa

leg_geo_qual

leg_longitude

leg_latitude

leg_flag_geo

es_state FIPS code of the state

1.1.7.10 10_fill_sync.sas

es_ein cleaned SEINUNIT EIN

9 digit federal firm identifier. Generally not unique within a state. There may be multiple state level firms for a given federal firm identifier.

es_county cleaned SEINUNIT county
3 digit FIPS county code.

es_naics cleaned SEINUNIT NAICS

es_owner_code cleaned SEINUNIT ownership code

1 = Federal Government

2 = State Government

3 = Local Government

5 = Private Sector

es_sic cleaned SEINUNIT SIC

es_(sic, naics, county, owner_code, ein)_miss 0 = Variable is not missing

1 = Variable is missing before using information from other quarters.

2 = Variable is not missing after search for off quarter information.

mode_es_XXX_emp+4 = Variable is missing, filled with the SEIN employment weighted mode value.

es_(sic, naics, county, owner_code, ein)_flag Missing = No information in other quarters

0 = Variable is not missing in current quarter

i0 = quarter after the current quarter where replacement value is found

j0 = quarter before the current quarter where replacement value is found

1.1.7.11 11_sein_mode_calc.sas

mode_(es_sic, es_naics, es_county, es_owner_code, es_ein, leg_wib, leg_msapmsa, leg_state, leg_county, leg_subctygeo) The modal value of the variable in an SEIN YEAR QUARTER (unit weighted)

mode_(es_sic, es_naics, es_county, es_owner_code, es_ein, leg_wib, leg_msapmsa, leg_state, leg_county, leg_subctygeo)_emp The modal value of the variable in an SEIN YEAR QUARTER (employment weighted)

1.1.7.12 12_sein_wide.sas

Place **SIC, NAICS, COUNTY, ownership code, EIN** and **LEG SEIN** level variables in arrays

1.1.7.13 13_sein_fill.sas

mode_es_(sic, naics, county, owner_code, ein)_miss 0 = Variable is not missing

1 = Variable is missing before using information from other quarters.

2 = Variable is not missing after search for off quarter information.

6 = Variable is missing, filled with imputed value. Currently only used for SIC.

11 = variable missing, but value set to 5. Currently only used for owner_code. Assume records with missing ownership codes are private firms.

mode_es_(sic, naics, county, owner_code, ein)_emp_miss 0 = Variable is not missing

1 = Variable is missing before using information from other quarters.

2 = Variable is missing, filled with off quarter information.

5 = Variable is missing, filled with the corresponding unit weighted value

6 = Variable is missing, filled with imputed value. Currently only used for SIC.

11 = variable missing, but value set to 5. Currently only used for owner_code. Assume records with missing ownership codes are private firms.

mode_es_(sic, naics, county, owner_code, ein)_flag Missing = No information in other quarters

0 = Variable is not missing in current quarter

¿0 = quarter after the current quarter where replacement value is found

¡0 = quarter before the current quarter where replacement value is found

1.1.7.14 14_impute_sein_industry.sas

SEIN mode variables missing values are replaced. Missing codes are adjusted. See program 13 for an explanation of valid values.

1.1.7.15 15_impute_seinunit_industry.sas

SEINUNIT mode variables missing values are replaced. Missing codes are adjusted. See program 10 for an explanation of valid values.

1.1.7.16 16_seinunit_yq_chars.sas

es_sic_div SIC divisions (A, B, C,, Z)

ES_SIC_2 First 2 digits of the 4 digit SIC

ES_SIC_3 First 3 digits of the 4 digit SIC

ES_NAICS_2 First 2 digits of the 6 digit NAICS

ES_NAICS_3 First 3 digits of the 6 digit NAICS

1.1.7.17 17_weight_calc_01.sas

Only temporary variables used in the calculation of the weights are created.

1.1.7.18 18_weight_calc_02.sas

qwi_unit_weight = Final ECF weight. See technical documentation for the weights for detailed information.

1.1.7.19 19_seinunit_fuzz.sas

Suppressed for confidentiality

1.1.7.20 20_sein_file.sas

No new variables are created.

1.1.7.21 21_fuzz_update.sas

DATE_(SEIN, SEINUNIT)_FUZZ SAS date value for when the fuzz factor was created.

UPDATE_NUMBER_(SEIN, SEINUNIT) Sequential update number. The first time the ECF is created all fuzz factors receive a value of 0. The value is incremented by 1 each time any fuzz factors are added to the master file.

1.2 NAICS codes on the ECF

Enhanced NAICS variables are available on all ECF since February 2003. The variable list below shows that there are 75 new variables for NAICS alone. The variables can be differentiated mainly by the source(s) and coding system used in their creation. There are two sources of data; the ES202 and the LDB from the BLS; and two coding systems; NAICS1997 and NAICS2002 (see the Census web site for more info.). Every NAICS variable uses at least one source and one coding system.

The ESO and FNL variables are of primary importance to the user community. The ESO variables use ONLY information from the ES202 and ignore any information that may be available on the LDB (see Section 1.2.1 for some analysis on why this may be preferred). The FNL variables incorporate information from both the ES202 and the LDB, with the LDB being the primary source. The ES_NAICS_FNL1997 and ES_NAICS_FNL2002 should be used to create the QWI estimates. Both the ESO and the FNL variables contain no missing values.

1.2.1 LDB versus LEHD NAICS backcoding

The LDB algorithm is to some extent a black box and testing has shown that it does a relatively poor job of capturing firm industry changes that occurred during the 1990's. In fact, the LDB appears to be a simple backfill that does not take into account a firm's entire SIC history.

Although some of the SIC changes over time may be spurious, a firm's SIC code history contains valuable information that we have attempted to preserve in our imputation algorithm. Overall, the effect of the different approaches is relatively small, since very few firms change industry, in particular relative to the proportion of firms that change geography.

In the following, we present a summary of research done on the ESO vs. FNL NAICS codes.

The NAICS_LDB variable is used for about 85% of the records for Illinois, the rest are filled with information from the ES202 (not sure why only 85% of the records on our ES202 files are in the LDB. The results weighted by employment are about the same suggesting that activity was not a criterion for being included on the LDB). First and not surprisingly, in later years and quarters (1999+) when NAICS is actively coded by the states, the codes look almost identical when available.

Second, there is little variation in the LDB NAICS codes over time compared with SIC. Among all of the active SEIN SEINUNITs over the period, a little over 8% experience at least one SIC change compared with about 1.5% on the LDB (almost all of these are 1999+). While this is not entirely unexpected, it is something to keep in mind when comparing NAICS_FNL versus SIC or NAICS_ESO employment totals. Many of these changes in industry appear to be real and are not captured on the LDB.

One effect of this is that as we go back in time a larger portion of employment can be found in NAICS_FNL codes that are different than one would expect given the SIC code on the ECF. For example, in 1990 about

13% of employment is in a NAICS.FNL code that is different than what we would expect based on the SIC. By 2001 this number falls to 3%. The ES202 based NAICS variable does a better job tracking SIC, since more SIC information is used in putting it together (about 3% consistently over the period).

The main source of the discrepancy is due to entities that experience a change in their SIC code prior to 2000. The LDB appears to ignore this change, while the ESO NAICS variable uses an SIC based impute for these SEINUNITS. The result is a series that exhibits similar patterns of change over time as SIC, while still preserving the value added in the NAICS codes for entities that did not experience a change.

Also, users should keep in mind that for early years (¡1997) some of the NAICS industries have yet to come into existence. I have no estimates on the prevalence of this problem.

1.2.2 Variable List

Variable Name	Source	Notes
es_naics_aux1997	ES202 NAICS AUX variable	BLS coding of aux estabs
es_naics_aux1997_flag		
es_naics_aux1997_miss		
es_naics_aux1997_src		
es_naics_aux2002		
es_naics_aux2002_flag		
es_naics_aux2002_miss		
es_naics_aux2002_src		
es_naics_eso1997	ES202 NAICS AUX, NAICS, SIC	Only ES202 info used
es_naics_eso1997_miss		
es_naics_eso1997_src		
es_naics_eso2002		
es_naics_eso2002_miss		
es_naics_eso2002_src		
es_naics_fnl1997	BLS LDB and ESO Input vars.	All industry info used
es_naics_fnl1997_2		
es_naics_fnl1997_3		
es_naics_fnl1997_4		
es_naics_fnl1997_5		
es_naics_fnl1997_miss		
es_naics_fnl1997_src		
es_naics_fnl2002	BLS LDB and ESO input vars.	All industry info is used
es_naics_fnl2002_2		
es_naics_fnl2002_3		
es_naics_fnl2002_4		
es_naics_fnl2002_5		
es_naics_fnl2002_miss		
es_naics_fnl2002_src		
es_naics_imp1997	ES202 SIC code	Impute using only SIC
es_naics_imp1997_miss		
es_naics_imp1997_src		
es_naics_imp2002		
es_naics_imp2002_miss		
es_naics_imp2002_src		
es_naics_ldb1997	BLS LDB NAICS variable	
es_naics_ldb1997_flag		

Variable Name	Source	Notes
es_naics_ldb1997_miss		
es_naics_ldb1997_src		
es_naics_ldb2002		
es_naics_ldb2002_flag		
es_naics_ldb2002_miss		
es_naics_ldb2002_src		
es_naics1997	ES202 NAICS Only	
es_naics1997_flag		
es_naics1997_miss		
es_naics1997_src		
es_naics2002		
es_naics2002_flag		
es_naics2002_miss		
es_naics2002_src		
mode_es_naics_eso1997	Mode of ESO SEINUNIT var	
mode_es_naics_eso1997_emp		
mode_es_naics_eso1997_emp_flag		
mode_es_naics_eso1997_emp_miss		
mode_es_naics_eso1997_flag		
mode_es_naics_eso1997_miss		
mode_es_naics_eso2002		
mode_es_naics_eso2002_emp		
mode_es_naics_eso2002_emp_flag		
mode_es_naics_eso2002_emp_miss		
mode_es_naics_eso2002_flag		
mode_es_naics_eso2002_miss		
mode_es_naics_fnl1997	Mode of FNL SEINUNIT var	
mode_es_naics_fnl1997_emp		
mode_es_naics_fnl1997_emp_flag		
mode_es_naics_fnl1997_emp_miss		
mode_es_naics_fnl1997_flag		
mode_es_naics_fnl1997_miss		
mode_es_naics_fnl2002		
mode_es_naics_fnl2002_emp		
mode_es_naics_fnl2002_emp_flag		
mode_es_naics_fnl2002_emp_miss		
mode_es_naics_fnl2002_flag		
mode_es_naics_fnl2002_miss		

1.2.3 Coding of MISS and SRC

Each new NAICS variable has several associated variables of which the miss and src variable are the most important.

1.2.3.1 MISS Variable Codes

If information from another period is used, the flag variable reports how many quarters away the NAICS value was found. Values greater than six should only appear in SEINUNIT level variables. If NAICS is missing for all quarters, then the SEINUNIT value has been filled with the SEIN value. The SEINUNIT codes represent the SEIN value +5.

Table 1.4: MISS Variable Codes

0	=	Valid value available in that period
1	=	Missing
1.5	=	(1999 and earlier only) Filled using impute based on SIC due to an SIC change over the period.
2	=	Filled using own code from another period
3	=	Filled from another source contemporaneously
5	=	Filled using the non-employ weight mode (SEIN mode var only)
6	=	Unconditionally imputed (SEIN mode var only)
6	=	NAICS imputed using SIC unconditional impute (SEIN mode var only)
7	=	Filled using the SEIN mode from another period (sic, fnl and eso vars only)
11	=	Filled using unconditional impute of SEIN value (sic, fnl and eso vars only)

1.2.3.2 SRC Variable Codes

The ESO and FNL variables use the following source codes. If more detail is desired about the source of the NAICS code, the user must look to the SRC code for that source. For example, if the ESO source code for ES_NAICS_ESO1997 says NCS, then the actual SRC information will be found in ES_NAICS1997_SRC.

Table 1.5: SRC Variable: ESO, FNL

AUX	=	Source is the ES202 NAICS AUX variable
LDB	=	Source is the LDB NAICS variable
NCS	=	Source is the ES202 NAICS variable
SIC	=	Source is the ES202 SIC code

The AUX, LDB and standard NAICS codes have the following source variables.

Table 1.6: SRC Variable: AUX, LDB, NAICS

SIC	=	Source is the ES202 SIC code
NO2	=	Source is a NAICS 2002 Code
N97	=	Source is a NAICS 1997 Code

1.2.4 NAICS algorithm precedence ordering

Four basic sources of industry information are available on the ECF; NAICS, NAICS_AUX, SIC, and the NAICS_LDB. The NAICS, NAICS_AUX, and NAICS_LDB missing values were filled using the following preference ordering. SIC is filled similarly, except miss=1.5 is not used and NAICS, not SIC, would be the basis for the impute when miss=3.

1. Valid 6 digit industry code (miss=0)
2. Imputed code based on first 3,4, or 5 digits when no valid six digit code is available in another period (miss=0)

3. Imputed code based on contemporaneous SIC if SIC changed prior to 2000 (miss=1.5)
4. Valid 6 digit code from another period (miss=2)
5. Valid code from another source (for example if NAICS1997 is missing, NAICS2002 or SIC may be available) (miss=3)
6. Use SEIN mode value (miss=5,7)
7. Unconditional impute (miss=6,11)

1.2.5 ESO and FNL variables

The ESO and FNL variables are made up of combinations of the various sources of industry information. The ESO variable uses the NAICS and NAICS_AUX variables as input. Information from the variable with the lowest MISS value is preferred although in case of a tie the NAICS_AUX value is used.

The FNL variable uses the ESO and LDB variables. Information from the variable with the lowest MISS value is preferred although in case of a tie the NAICS_LDB value is used. Keep in mind that although the source of an ESO or FNL variable may be equal to NCS, the actual source can only be ascertained by going back to the original.

1.2.6 Employment Flag Variable Codes

All current uses of the ECF have been forced to assume that employment and payroll information has been reported by the firm, although under certain conditions the ES202 processing specifications require imputation of missing values. The flag values below allow the user to determine when imputation has occurred.

The master record contains valuable information that has been preserved in the master_empl_month1_flg –master_total_wages_flg variables. For example, one should theoretically be able to distinguish 0 prorated codes from 0 unknowns by looking at multi units with masters that reported (code=1) and subunits with a zero.

The following information stems from an email exchange between Kevin McKinney (U.S. Census Bureau) and George Putnam (Illinois) on 12/15/2003.

Prior to late 1995:

0 = unknown

1 = not imputed

2 = imputed (including prorated multiple worksite data)

Late 1995 or early 1996:

0 = prorated data (multiple worksites)

1 = actual or not imputed data

2 = estimated data

1997 first quarter forward (ES202 processing manual, Appendix B):

- Blank* = *reported data*
- R* = *reported data*
- A* = *estimated from CES report*
- C* = *changed (re-reported)*
- D* = *reported from missing data notice*
- E* = *imputed single unit employment or imputed worksite employment
prorated from imputed parent record*
- H* = *hand-imputed (not system generated)*
- L* = *late reported (overrides prior imputation)*
- M* = *missing data*
- N* = *zero-filled pending resolution of long-term delinquent reporter*
- P* = *prorated from reported master to worksite*
- S* = *aggregated master from reported MWR or EDI data*
- W* = *estimated from wage record employment*
- X* = *non-numeric employment zero-filled pending further action*

1.2.7 Multi-Unit Code or MEEI

The MULTI_UNIT variable on the ECF is determined by counting the number of SEINUNIT records for a given SEIN once the master records have been removed. However, some multiunit firms refuse to report detailed information for their sub-units and appear as single units on the ECF. The table below provides an estimate of the magnitude of multiunit firms refusing to report detailed unit information using data from Illinois.

MULTI_UNIT_CODE	MULTI_UNIT	
	0	1
1	1,483,808	0
2	1	0
3	120	155859
4	5808	0
5	0	33
6	13899	0

Prior to 1997 (ES202 processing manual sent from George Putnam):

- 1 = Single establishment unit
- 2 = Multi-unit master record
- 3 = Subunit establishment level record for a multi-unit employer
- 4 = Multi-establishment employer reporting as a single unit due to unavailability of data, including refusals
- 5 = A subunit record that actually represents a combination of establishments; finer level breakouts are not yet available
- 6 = Known multi establishment employer reporting as a single unit and not solicited for disaggregation because of small employment (< 10) in all secondary establishments combined

1997 first quarter forward (ES202 processing manual, Appendix B):

- 1 = Single establishment unit
- 2 = Multi-unit master record
- 3 = Subunit establishment level record for a multi-unit employer
- 4 = Multi-establishment employer reporting as a single unit due to unavailability of data, including refusals
- 5 = A subunit record that actually represents a combination of establishments; finer level breakouts are not yet available
- 6 = Known multi establishment employer reporting as a single unit and not solicited for disaggregation because of small employment (j 10) in all secondary establishments combined

1.2.8 Auxiliary Code

This variable gives detailed information about firm locations that do not directly engage in production related activities.

Prior to 1997 (ES202 processing manual sent from George Putnam):

- 0 = Unknown
- 1 = Central administrative office
- 2 = Performs research, development or testing services
- 3 = Provides storage or warehouse services
- 5 = Does not provide auxiliary services, it is an operating establishment
- 9 = Performs auxiliary services that are not described above

1997 first quarter forward (ES202 processing manual, Appendix B):

- 0 = Auxiliary status not known
- 1 = Central administrative office
- 2 = Performs research, development or testing services
- 3 = Provides storage or warehouse services
- 5 = Does not provide auxiliary services, it is an operating establishment
- 6 = Headquarters
- 7 = Administrative, Other than Headquarters
- 9 = Performs auxiliary services that are not described above

1.3 History and directions for the future

1.3.1 History

Version 3.1 retains a large portion of the version 2.2 and version 3.0 programs, while at the same time expanding the ECF in several areas. The sections of code that determine the record structure and calculate various employment and payroll measures remain largely unchanged from version 2.2. However, new sections of code were added in version 3.0 that integrate detailed geography data from the LEG and substantially improve the quality of industry coding. The fuzz section was also rewritten in version 3.0. The multiple conditional fuzz distributions have been replaced with a single distribution that no longer depends on the number of firms in a county industry cell. Version 3.1 integrates the calculation of the weights into the main program sequence (the weights were a separate patch in version 3.0). Finally, the fuzz factor programs were modified to allow the retention of existing fuzz factors when a new ECF is created. New fuzz factors for an SEIN SEINUNIT are only created when a record does not already exist.

The LEG was integrated into the ECF jobstream (as of 3.1.13).

Version 3 of the GAL was integrated into the ECF jobstream as of 3.1.32.

More details can be found in the OTHER subdirectory of the ECF code library (/programs/production/code_version/ecf) directories.

1.3.2 Directions for the Future

- Use the Business Register to augment missing data (may not be possible due to IRS restrictions).
- Add indicators for a link between the ES202 and Census Business Data (may not be possible due to IRS restrictions).
- Fix records that are almost duplicates. For example, the last digit of an SEIN for many states is not significant. Firms that have the same first 11 digits are often related in some way. Eventually, these records were either consolidated or eliminated in the mid 1990's. Successor / Predecessor work may identify many of these firms.
- Complete NAICS code cleanup and imputation. [DONE as of 3.1.14]
- Change the code to allow for updates without starting the complete ECF sequence from the beginning every time. Most of the ECF can be run autonomously when a new quarter is received except for two notable exceptions; 1. The record structure impute requires data from earlier quarters, although prior quarters' data is not subject to modification. 2. The filling of missing SIC, NAICS, EIN, etc. requires information from earlier quarters and all data is subject to modification (old reports can fill missing values in newly received data and new reports can fill missing values in old data).
- Multi-unit indicator for multi-units that report as single units
- Use the employment/total wage flags to distinguish between reported and imputed data on the ECF. This will be useful for the weights and other research uses in general.

1.4 How to run diagnostics

The ECF traditionally has had a very compute-intensive diagnostic process, generating a very large list file. Since version 3.1.48, this has been moved to a separate QA process, which is only run on-demand. This section describes how to generate those diagnostics.

1.4.1 When to run the diagnostics

The diagnostics should only be run *before* QA signoff. After QA signoff, it is likely that a lot of the intermediate data files, kept in the INTERWRK library, have been deleted.

1.4.2 Modifying the parameters file

In the runtime directory, edit 'parameters.ecf.sas'. Line 20 should contain a line reading

```
%let diagnostics=no;
```

Changing this parameter will enable diagnostics.

1.4.3 Which file to run

Once the parameter has been modified, do *not* re-run the vintage runtime file for the main process (typically called 'ecf_[state]_[vintage].sas'). Rather, run the vintage *QA* runtime file, typically called

```
ecfqa_[state]_[vintage].sas
```

Then have a look at the generated list file. This file is typically in the multiple GB range, so be careful how you open it.

1.5 Details

1.5.1 Control programs

Control programs are used to setup and manage modules, define the sequence in which modules are run, and provide for hooks to QA and control database updates.

1.5.1.1 ctrlprogs/control_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: ctrlprogs/control_ecf.sas >--+ */
/* file: control_ecf.sas */

%let this_process=ecf;
%let debug=yes;
%let start_year=;
%let start_quarter=;
%let end_year=;
%let end_quarter=;
/*-----*/
```

Tips on running the ECF/LEG sequence.

We condition on all available years of ES202 data. This defines the year range of the ECF. Conditional on having defined the ES202 years, a GAL cross-walk should exist for every one of those years, a fake xwalk is automatically created otherwise. Missing GAL xwalks are allowed to occur at the beginning and/or the end of the date range. No holes in the range are allowed, nor will they occur under normal circumstances.

When incorporating the LEG, we really need no longer produce the LEG, nor the LEG structure. However, until we figure out if it is actually needed, we still keep it in here. Hard coded reference to gal year in order to take into account the new GAL 3 naming conventions

```

%let gal_geocode_year=2003;
/*=====*/
/*----- generate list of required files ----- */
/*=====*/

data required_files;
  length process subunit $ 20 table $ 50 state $ 2 fail 3;
  input process subunit fail;
  keep table fail;
  if process = "parms" then state="us";
  else if process = "gal" and subunit = "subcty" then state="us";
  else state="&state.";
  if ( subunit = "" ) then table=trim(process)||"_"||trim(state);
  else table=trim(process)||"_"||trim(state)||"_"|| trim(subunit);
  /* construct yearly files */
  if process = "es202" or process= "ldb" then do;
do year=1985 to 2015;
  /*---- table=trim(process)||"_"||trim(state)||"_"||trim(left(year));
---*/
  /*---- This part will become active when the ES202 goes quarterly
----*/
  do quarter= 1 to 4;

table=trim(process)||"_"||trim(state)||"_"||trim(left(year))||"q"||trim(left(qua
rter));
  output;
  end;
  /* end quarter loop */
  end; /* end year loop */
end; /* end es202 */
  else if ( process = "gal" and subunit = "xwalk" ) then do;
do year=1985 to 2015;
table=trim(process)
  ||"_"
  ||trim(state)
  ||"_"&gal_geocode_year."_
  || trim(subunit)
  || "_"
  ||trim(left(year));
output;
  end; /* end year loop */
  end;
  else if ( process = "gal" and subunit = "core" ) then do;

table=trim(process)
  ||"_"
  ||trim(state)
  ||"_"&gal_geocode_year.";
output;

  end;

  else if ( process = "gal" and subunit = "tccb" ) then do;

table=trim(process)
  ||"_"
  ||trim(state)
  ||"_"&gal_geocode_year."_tccb";
output;

  end; /* end gal */

  else output;
  /* old files pre-LEG merge */
  /* leg . 0 */

```

This lists all required files.

```
datalines;
  es202 . 0
  ldb . 0
  ecf sein_fuzz 0
  ecf seinunit_fuzz 0
  ehf sein_employment 1
  ehf controltotals 1
  gal core 1
  gal tccb 0
  gal xwalk 0
  parms wib_subcty 1
  parms imp_sic_sic3 1
  parms imp_sic_sic2 1
  parms naics_imp_ncs1997_ncs2002 1
  parms naics_imp_ncs2002_ncs1997 1
  parms naics_imp_ncs1997_sic 1
  parms naics_imp_sic_ncs1997 1
  parms fuzz 1
;;
run;
proc print data=required_files;
run;
```

The WIB.US.SUBCTY file is occasionally updated by the GAL/WIB process/person, but should always be present. If the file is outdated and missing a new state, the runtime process will fail with an error message.

```
/*----- find required processes -----*/
%macro
require_input_files(input=required_files,output=vintage_input_tables,label=REQUI
RE_INPUT_FILES);

proc sort data=&input.;
by table;
run;

proc sort data=available_tables;
by table;
run;

data _required _missing _noncrucial;
merge &input.(in=a) available_tables(in=b);
by table;
if qa_flag>0
and current_flag=1
;
if a and not b and fail then output _missing;
if a and not b and not fail then output _noncrucial;
if a and b then output _required;
run;
proc contents data=available_tables;
run;
```

```

data _null_;
  missid=open("_missing");
  missnobs=attrn(missid,"nobs");
  call symput("missnobs",trim(left(missnobs)));
  rc=close(missid);

  noncrucialid=open("_noncrucial");
  noncrucialnobs=attrn(noncrucialid,"nobs");
  call symput("noncrucialnobs",trim(left(noncrucialnobs)));
  rc=close(noncrucialid);

  reqid=open("_required");
  reqnobs=attrn(reqid,"nobs");
  call symput("reqnobs",trim(left(reqnobs)));
  rc=close(reqid);
run;

%if ( &noncrucialnobs > 0 ) %then %do;
  %put +++++ &label. %upcase(warning) +++++ ;
  %put +++++ &label. %upcase(warning) +++++ ;
  %put +++++ &label. %upcase(warning) +++++ ;
  %put +++++ &label. %upcase(warning) +++++ The following non-crucial
file(s);
  %put +++++ &label. %upcase(warning) +++++          could not be
found;
  %put +++++ &label. %upcase(warning) +++++ ;
  %put +++++ &label. %upcase(warning) +++++ ;
  data _null_;
    set _noncrucial;
    put "+++++ &label. %upcase(warning) +++++ MISSING: " table;
  run;
%end; /* end noncrucial condition */
%if (&missnobs = 0) %then %do;
  proc append base=vintage_input_tables data=_required;
  run;
%end;

%else %do;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ Could not find a required
file.;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ ;
  data _null_;
    set _missing;
    put "+++++ &label. %upcase(error) +++++ MISSING: " table;
  run;
  %goto fault;
%end; /* end fault condition */
/*----- prepare to exit either with error or without -----*/
%goto endall;

```

```

%fault:
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ An error has occurred.
Please verify ;
  %put +++++ &label. %upcase(error) +++++ Exiting.;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ ;
  %put +++++ &label. %upcase(error) +++++ ;
  endsas;

%endall:
  %put +++++ &label. %upcase(info) +++++ ;
  %put +++++ &label. %upcase(info) +++++ ;
  %put +++++ &label. %upcase(info) +++++ ;
  %put +++++ &label. %upcase(info) +++++ The following files were
found, and will be used.;
  %put +++++ &label. %upcase(info) +++++ Verify that this is
correct.;
  %put +++++ &label. %upcase(info) +++++ ;
  %put +++++ &label. %upcase(info) +++++ ;
  data _null_;
    set vintage_input_tables;
    put "+++++ &label. %upcase(info) +++++ USING : " table;
  run;

%mend; /*===== END OF REQUIRE MACRO
=====*/

```

The big Kahuna.

```
%macro control_ecf;
```

Here we request all required files. SAS will exit with an error message if not all files are found. Input by default is required_files.

```

%require_input_files(label=%sysfunc(upcase(control_&this_process)));

/*=====*/

```

Now we figure out the maximum range of the ES202, and check that we have a GAL xwalk for each year of the Es202. We also check whether there is a GAL xwalk for every year of the ES202 (THIS MAY NOT BE NECESSARY)

```

/*=====*/

proc sql;
create table es202range as
  select *
  from vintage_input_tables
  where process_id='es202'
     and qa_flag>0
     and current_flag=1
     and state="&state"
  order by start_year, start_quarter, end_year, end_quarter
  ;
create table galrange as
  select *
  from vintage_input_tables
  where process_id='gal'
     and index(table,"gal_&state._gal_geocode_year._xwalk") ne 0
     and qa_flag=1
     and current_flag=1
     and state="&state"
  order by start_year, start_quarter, end_year, end_quarter
  ;
quit;

%if (&debug ne ) %then %do;
  proc print data=es202range;
  run;
  proc print data=galrange;
  run;
%end;

```

Joint availability: There should be no ONLYGAL, since the GAL depends on the ES202, but we check.

The ES202 is quarterly files, the GAL are yearly files. We thus first aggregate up the quarterly files to their year range.

```

data yearly_es202;
  set es202range
(rename=
  (start_year=s_year
    start_quarter=s_quarter
    end_year=e_year
    end_quarter=e_quarter));
  by s_year s_quarter;
  drop s_: e_: table;
  retain start_year start_quarter end_year end_quarter;
  if first.s_year then do;
start_year=s_year;
start_quarter=s_quarter;
end;
  if last.s_year then do;
end_year=e_year;
end_quarter=e_quarter;
output;
end;
  run;

data joinrange _missinggal _onlygal;
  merge
yearly_es202(in=a)
galrange(in=b)
;
  by start_year start_quarter end_year end_quarter;
  if a and b then output joinrange;
  if a and not b then output _missinggal;
  if b and not a then output _onlygal;
run;

data _null_;
  missid=open("_missinggal");
  missnobs=attrn(missid,"nobs");
  call symput("missnobs",missnobs);
  rc=close(missid);

  onlygalid=open("_onlygal");
  onlygalnobs=attrn(onlygalid,"nobs");
  call symput("onlygalnobs",onlygalnobs);
  rc=close(onlygalid);
run;

```

Now put diagnostic output to the log. Note that 'lehd_control' parses for upper-case warning, and thus should print this to the screen when running the control job.

```

%if ( &onlygalnobs > 0 ) %then %do;
  %let label=%upcase(warning);
  %put +++++ &label. +++++ ;
  %put +++++ &label. +++++ ;
  %put +++++ &label. +++++ ;
  %put +++++ &label. +++++ %upcase(warning): There seems to be a GAL
xwalk and no ES202.;
  %put +++++ &label. +++++          This situation should not occur.
Please check.;
  %put +++++ &label. +++++ ;
  %put +++++ &label. +++++ ;
  data _null_;
    set _onlygal;
    put "+++++ &label. +++++ ONLYGAL: " table;
  run;
  /* endsas; */
%end;

```

End onlygal condition

```

%if ( &missnobs > 0 ) %then %do;
%let label=%upcase(warning);
%put +++++ &label. +++++ ;
%put +++++ &label. +++++ ;
%put +++++ &label. +++++ ;
%put +++++ &label. +++++ %upcase(warning): A GAL xwalk was not
found for certain years.;
%put +++++ &label. +++++ of ES202 data. The ECF (LEG)
will build fine,;
%put +++++ &label. +++++ but this might be a problem;
%put +++++ &label. +++++ ;
%put +++++ &label. +++++ ;
data _null_;
set _missinggal;
put "+++++ &label. +++++ MISSING GAL for: " table;
run;
/* endsas; */
%end; /* end onlygal condition */

```

Computing start and end date range for ES202

```

data es202range
(keep=process_id start_year start_quarter end_year end_quarter
)
;
set es202range(rename=(
start_year =old_start_year
start_quarter=old_start_quarter
end_year =old_end_year
end_quarter =old_end_quarter
)
)
end=eof;
if _N_ =1 then do;
start_year =old_start_year ;
start_quarter=old_start_quarter;
end;
if eof then do;
end_year =old_end_year ;
end_quarter =old_end_quarter ;
output;
call symput('es202_start_year',trim(left(start_year )));
call symput('es202_start_quarter',trim(left(start_quarter)));
call symput('es202_end_year',trim(left(end_year )));
call symput('es202_end_quarter',trim(left(end_quarter )));
end;
retain start_year start_quarter end_year end_quarter;
run;

```

Computing start and end date range for GAL

```

%let gal_start_year=0;
%let gal_start_quarter=0;
%let gal_end_year=0;
%let gal_end_quarter=0;

data galrange
  (keep=process_id start_year start_quarter end_year end_quarter
  )
  ;
  set galrange(rename=(
    start_year =old_start_year
    start_quarter=old_start_quarter
    end_year =old_end_year
    end_quarter =old_end_quarter
  )
  )
end=eof;
  if _N_ =1 then do;
    start_year =old_start_year ;
    start_quarter=old_start_quarter;
  end;
  if eof then do;
    end_year =old_end_year ;
    end_quarter =old_end_quarter ;
  output;
    call symput('gal_start_year',trim(left(start_year )));
    call symput('gal_start_quarter',trim(left(start_quarter)));
    call symput('gal_end_year',trim(left(end_year )));
    call symput('gal_end_quarter',trim(left(end_quarter )));
  end;
  retain start_year start_quarter end_year end_quarter;

run;

```

Computing start and end date range for EHF

```

data ehfrange
  (keep=process_id start_year start_quarter end_year end_quarter
  )
  ;
  set vintage_input_tables(where=(process_id="ehf"))
end=eof;
  if eof then do;
    call symput('ehf_start_year',trim(left(start_year )));
    call symput('ehf_start_quarter',trim(left(start_quarter)));
    call symput('ehf_end_year',trim(left(end_year )));
    call symput('ehf_end_quarter',trim(left(end_quarter )));
  end;
run;

%if (&debug ne ) %then %do;
proc print data=es202range;
  run;
proc print data=galrange;
  run;
proc print data=ehfrange;
  run;
%end; /* end debug condition */

```

Check for valid dates, and fail otherwise.

```

%if ( /* ES202 conditions */
      (&es202_start_year >= 1985 )      and      (&es202_start_year < 2050 )
      and
      (&es202_start_quarter >= 1 )      and      (&es202_start_quarter <= 4 )
      and
      (&es202_end_year >= 1985 )      and      (&es202_end_year < 2050 )
      and
      (&es202_end_quarter >= 1 )      and      (&es202_end_quarter <= 4 )
      and
      (&es202_end_year > &es202_start_year )
and
  /* GAL conditions */
  (&gal_start_year >= 1985 )      and      (&gal_start_year < 2050 )
  and
  (&gal_start_quarter >= 1 )      and      (&gal_start_quarter <= 4 )
  and
  (&gal_end_year >= 1985 )      and      (&gal_end_year < 2050 )
  and
  (&gal_end_quarter >= 1 )      and      (&gal_end_quarter <= 4 )
  and
  (&gal_end_year > &gal_start_year )
and
  /* EHF conditions */
  (&ehf_start_year >= 1985 )      and      (&ehf_start_year < 2050 )
  and
  (&ehf_start_quarter >= 1 )      and      (&ehf_start_quarter <= 4 )
  and
  (&ehf_end_year >= 1985 )      and      (&ehf_end_year < 2050 )
  and
  (&ehf_end_quarter >= 1 )      and      (&ehf_end_quarter <= 4 )
  and
  (&ehf_end_year > &ehf_start_year )
)
%then %do;
  %put ECF %upcase(warning): ECF starts in
&es202_start_year.:&es202_start_quarter;
  %put ECF %upcase(warning): ECF ends in
&es202_end_year.:&es202_end_quarter;
  %put ECF %upcase(warning): GAL starts in
&gal_start_year.:&gal_start_quarter;
  %put ECF %upcase(warning): GAL ends in &gal_end_year.:&gal_end_quarter;
  %put ECF %upcase(warning): EHF starts in
&ehf_start_year.:&ehf_start_quarter;
  %put ECF %upcase(warning): EHF ends in &ehf_end_year.:&ehf_end_quarter;
%end;
%else %do;
  %put ECF %upcase(error): Invalid start or end dates for EHF, ES202, or GAL;
  %put ECF %upcase(error): ECF will exit. No vintage created.;
endsas;
%end;

/*=====
*/

```

Now we define the output files.

```

/*=====
*/
    data output_tables;
    length out_table $50. ;
    i=1;
    ot='out' || trim(left(i));
    out_table="(&this_process._&state."
        || "_sein"
        || ")";
    output;
    call symput(ot,out_table);
    i+1;
    ot='out' || trim(left(i));
    out_table="(&this_process._&state."
        || "_seinunit"
        || ")";
    output;
    call symput(ot,out_table);

    call symput(ot,out_table);
    i+1;
    ot='out' || trim(left(i));
    out_table="(&this_process._&state."
        || "_sein_fuzz"
        || ")";
    output;
    call symput(ot,out_table);
    i+1;
    ot='out' || trim(left(i));
    out_table="(&this_process._&state."
        || "_seinunit_fuzz"
        || ")";
    output;
    call symput(ot,out_table);

```

These last two tables might no longer be necessary

```

/* i+1;
* ot='out' || trim(left(i));
* out_table="(&this_process._&state."
*     || "_leg_structure"
*     || ")";
* output;
* call symput(ot,out_table);
*/
i+1;
ot='out' || trim(left(i));
out_table="(&this_process._&state."
    || "_leg"
    || ")";
output;
call symput(ot,out_table);
call symput('number_out',trim(left(i)));
run;

%let today=%sysfunc(today());
%let thisyear=%sysfunc(year(&today));

%put THE FOLLOWING ARE THE OUTPUT TABLES THAT WILL BE ASSOCIATED WITH THE NEW
VINTAGE ;
%do i=1 %to &number_out;
%put Output table : &&out&i;
%end;

```

This is a sanity check on the ymin and ymax parameters.

```

%let ymin=1985;
%let ymax=&thisyear;
%if ( &es202_start_year lt &ymin ) %then %let ymin=&es202_start_year;
%if ( &es202_end_year gt &ymax ) %then %let ymax=&es202_end_year;

```

Now go to BUILD_VINTAGE.

```

%if (&debug ne) %then %do;
data _null_;
  set vintage_input_tables;
  put "++++ DEBUG ++++ vintage_input_tables: " table;
run;
%end;

%build_vintage(process_id=&this_process.,
               current_state=&state,
               start_year=&es202_start_year.,
               start_quarter=&es202_start_quarter.,
               end_year=&es202_end_year.,
               end_quarter=&es202_end_quarter.,
               output_file=
                 %do i=1 %to &number_out;
                   &&out&i
                 %end;
               );

```

We now have access to INTERWRK, and we write out the different date range calculations for inclusion in the QA. Note that the running of the build_vintage macro will put all directories into place, so interwrk will exist, even though a runtime program has not yet been run. build_vintage will also create the interwrk libname.

```

data INTERWRK.es202range(
  label="This defines the year range of the ECF"
);
set es202range;
run;
data INTERWRK.galrange(
  label="This is the year range of the GAL"
);
set galrange;
run;
data INTERWRK.ehfrange(
  label="This is the year range of the EHF"
);
set ehfrange;
run;
/* this should be the same as for ES202 overall */
data INTERWRK.joinrange(
  label="This is the joint availability of ES202 and GAL"
);
set joinrange;
run;

data INTERWRK.dateranges;
set output_tables(rename=(out_table=table));
start_year=&es202_start_year.;
start_quarter=&es202_start_quarter.;
end_year=&es202_end_year.;
end_quarter=&es202_end_quarter.;
run;

```

Finally, we append some stuff to the parameters file. Some of this is legacy-compatibility and will not survive a more detailed code review.

```

data _null_;
  file
"&execute_root/&process_id/&state/&vintage/parameters_&this_process..sas" mod;
put ;
put " /*=====";
put "   Code generated by the control_&this_process. program for vintage
&vintage only";
put " =====*/";
put ;
put "*****";
put "* Assign Macro values";
put "*****";
put "%nrstr(%%)let gal_start_year=&gal_start_year; * gal start year;";
put "%nrstr(%%)let ehf_start_year=&ehf_start_year; * ehf start year;";
put "%nrstr(%%)let es202_start_year=&es202_start_year; * es start year;";
put "%nrstr(%%)let gal_end_year=&gal_end_year; * gal end year;";
put "%nrstr(%%)let ehf_end_year=&ehf_end_year; * ehf end year;";
put "%nrstr(%%)let es202_end_year=&es202_end_year; * es end year;";
put ;
put "%nrstr(%%)let ymin=&ymin; * earliest available year for any state data;";
put "%nrstr(%%)let ymax=&ymax; * latest available year for any state data;";
put ;
put '%let qbeg=%eval((&start_year.-&ymin.)*4+&start_quarter.); * earliest
sequential quarter;';
put '%let qend=%eval((&end_year.-&ymin.)*4+&end_quarter.); * latest
sequential quarter;';
put '%let qmin=1;';
put '%let qmax=%eval((&ymax-&ymin)*4+4);';
put '/*----- These are used by the LEG part -----*/';
put '%let leg_qrange=%eval((&end_year.-&start_year.+1)*4);';
put ;
put '/*---- other legacy ES202 names ----*/';
put '%let stfips=%sysfunc(stfips(&state),z2.);';
put ;
put '/* should be eliminated */';
put '%let einavail=ein; * ein if EIN is available, Empty string if not avail;';
put '/* should be changed to print to INTERWRK */';
put '%let listsize=10000; * Maximum number of records for long print
statements;';
put "%nrstr(%%)let gal_geocode_year=&gal_geocode_year; * GAL 3 geocode version
year;";

%mend;
%control_ecf;

```

1.5.1.2 ctrlprogs/parameters_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: ctrlprogs/parameters_ecf.sas >---+ */

*****;
**** Parameters ****;
*****;

```

This can be set at runtime to generate much more detailed diagnostics in the QA list file. It will print out all diagnostics previously scattered throughout the normal modules if set to 'YES' or 'yes', to the QA runtime list file, not the normal runtime list file.

```
%let diagnostics=no;

/*----- ex-ECF part -----*/
```

This used to be hardcoded, but was moved to a dataset in PARM5, and is read in just before assignment of fuzz factors.

```
*****
DO NOT CHANGE IF YOU DO NOT KNOW WHAT YOU ARE DOING!!!!
*****;

%let ltrim=.7; *lower bound for weight;
%let utrim=1.3; *upper bound for weight;

*****
* Assign Macro values
*****;

/*---- legacy names ----*/

%let curdir=%sysget(PWD);

%let stfips=%sysfunc(stfips(&state));

/* should be eliminated, since all states should have EIN defined on UI/ES202 */
%let einavail=ein; * ein if EIN is available, Empty string if not avail;
/* should be changed to print to INTERWRK */
%let listsize=10000; * Maximum number of records for long print statements;

/*----- ex-LEG part -----*/
```

Avoiding spikes, this is the cut-off value. If a unit's employment relative to county-wide employment is larger than this value, an effort is made not to assign that unit to this county. Introduced in 3.1.26.

```
%let leg_spike_cutoff=20;
```

Choosing the GAL version. It's either 1, 2 or 3.

```
%let gal_version=3;
```

Identify the preferred sub-county variable. This is maintained by the GAL team (Marc).

```

data _null_;
  set INPUTS.parms_us_wib_subcty end=eof;
  retain flag;
  if _n_=1 then flag='Not defined';
  if state="&state." then do;
    call symput('subcty',subcty);
    flag='Found';
  end;
  if eof and flag='Not defined' then do;
    put "%upcase(error): SUBCTY definition for &state " flag;
    call execute('endsas;');
  end;
run;
%put Selected Subcty as &subcty.;

```

Once the PRODUCTION format is working, this should be automatic.

```

%include "/programs/projects/auxiliary/Formats/format_&state..sas";
%include "/programs/projects/auxiliary/Formats/format_sic_division.sas";

%include "/programs/projects/auxiliary/Formats/multiformat_naics.sas";

%include "/programs/projects/auxiliary/Formats/multiformat_sic.sas";

```

1.5.2 SAS programs used

Modules can be either SAS programs that are '%include' in the main control program, or SAS macros that are made available by a '%include' or by a system-wide macro library, and executed by the control program. The preferred method are macros, but legacy programs use SAS programs as well.

1.5.2.1 library/sasprogs/01_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/01_ecf.sas >--+ */
/*----- LEGACY NAME: 01_read_es202.sas -----*/
```

This program reads in and stacks state ES 202 data. The first step is to clean up and standardize the data that is read in. I allow both state specific and general standardization rules. Any state specific data cleanup rules are placed in config_param.sas and the generic rules are contained in 01_read_all.sas. The idea is that all modifications are made to config_param.sas and none of the actual programs should have to be modified.

A large portion of the data quality checking now occurs in the ES202 read in before the data ever reaches the ECF. Some of the error checking code could probably be removed from the ECF at some point in the future.

I check the following variables: SIC, NAICS, county, SEIN, and EIN.

- SIC missing and 0000 are coded to 9999
- NAICS missing and 000000 are coded 999999
- NAICS_AUX missing and 000000 are coded 999999
- County missing, 000, 900,994,995,996,998 are coded to 999
- SEIN and SEINUNIT are checked for characters other than 0-9. Some SEIN's contain letters and the variable sein_bad=1 allows you to determine when that is the case (these SEIN's are often related to another SEIN in the data).
- Duplicates are checked. They should be removed during ES202 processing.
- EIN is also checked for characters other than 0-9. Letters and other ASCII codes are not allowed for the EIN. If any of these characters are encountered then ein_bad=1.
- I also check to see if the first two digits of the EIN correspond with a valid 2 digit IRS Revenue district code. The current list of codes is derived from the SSEL and information from the IRS. The SSEL contains clean EIN's to the best of my knowledge. EIN's on our state data that are a member of the set of 2 digit IRS Revenue district codes have a positive probability of matching to the SSEL. This list may need to be updated as the IRS adds additional codes over time. I also create an ein_defect variable that indicates what problems if any are found with the EIN.

```
/* ALL macros are stored in library/macros */

/*****
/* REQUIRES-DATA: yearly ES202 files from /data/master/ui/ss/YYYY/sasdata */
/* file name format is SSesYYYY.sas7bdat */
/* PROVIDES-DATA: ecf_stacked_01.sas7bdat */
*****/

title1 "Read in the ES202 from /data/master/ui/&state./YYYY";
title2 "Program Location: &curdir./01_read_es202.sas";
```

BEGIN DATA READ IN.

```

data
  INTERWRK.ecf_stacked_01
    (drop=temp_char i);
  %set_es; /* modified in 3.1.07 LV */
  by sein year quarter seinunit;

  length temp_char temp_ein $9 ein_char2 $2 county $3;

```

Generic Cleanup

```

county=put(county_temp, $cty_tmp.);

if owner_code in("4","6","7","8") then owner_code="5";
if owner_code in(" ", "0") then owner_code="9";
if naics=" " then naics="999999";
if naics="000000" then naics="999999";
if naics_aux=" " then naics_aux="999999";
if naics_aux="000000" then naics_aux="999999";
if sic=" " then sic="9999";
if sic="0000" then sic="9999";
if county=" " then county="ZZZ";
if county="000" then county="ZZZ";
if county="900" then county="ZZZ";
if county="994" then county="ZZZ";
if county="995" then county="ZZZ";
if county="996" then county="ZZZ";
if county="998" then county="ZZZ";
if county="999" then county="ZZZ";

```

Clean Up Payroll and Employment

```

if not empl_month1>0 then empl_month1=0;
if not empl_month2>0 then empl_month2=0;
if not empl_month3>0 then empl_month3=0;
if not total_wages>0 then total_wages=0;

```

State specific data clean up.

```

%state_specific_cleanup;

```

Check for Bad SEIN data.

```

sein_bad=0;
do i=1 to 12;
  temp_char=substr(sein,i,1);
  if temp_char<"0" or temp_char>"9" then sein_bad=1;
end;

```

Check for Bad SEINUNIT data.

```

seinunit_bad=0;
do i=1 to 5;
  temp_char=substr(seinunit,i,1);
  if temp_char<"0" or temp_char>"9" then seinunit_bad=1;
end;

```

Check for Duplicate SEIN YEAR QUARTER SEINUNIT records. This identifier should be unique.

```

duprec=0;
if first.seinunit~=1 or last.seinunit~=1 then duprec=1;

```

Check for Bad EIN data.

```

%check_ein(ein);

```

now output the data.

```

output INTERWRK.ecf_stacked_01;
run; /* end of readin */

proc sort data=INTERWRK.ecf_stacked_01;
by sein year quarter seinunit;
run;

/*===== Diagnostic Output =====*/

```

Moved to ecfdiag_01.sas

1.5.2.2 library/sasprogs/02_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* ---< Location: /programs/production/dev1/current/ecf >--+ */
/* ---< File: library/sasprogs/02_ecf.sas >--+ */
/*----- LEGACY NAME: 02_naics_clean.sas -----*/
/*****
/* REQUIRES-DATA: ecf_stacked_01.sas7bdat */
/* PROVIDES-DATA: naics_clean_02.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Clean and Standardize NAICS codes";
title2 "Program Location: &curdir./02_naics_clean.sas";

```

Collapse to the set of records with POTENTIALLY valid industry codes .

```

proc sort data=interwrk.ecf_stacked_01
(keep=sein year quarter seinunit naics where=(naics~="999999")) out=temp1;
by sein seinunit naics year quarter;
run;

```

Collapse to the set of unique industry codes for that SEIN SEINUNIT .

```

proc sort data=temp1(keep=sein seinunit naics) out=temp2 nodupkey;
by sein seinunit naics;
run;

```

Clean each industry code . Count the number of times a valid code is found .

```

data temp3(drop=naics_1997_valid naics_2002_valid) temp4(keep=sein seinunit
naics_1997_valid naics_2002_valid);
  set temp2;
  by sein seinunit;

  length naics_1997_invalid naics_1997_valid $1 naics_1997_format $7
         naics_2002_invalid naics_2002_valid $1 naics7 naics_2002_format $7;

  retain naics_1997_valid naics_2002_valid;

  /* xxx_valid is whether a valid code is found anywhere during the period */
  /* xxx_invalid is whether a valid code is present in the current period */

  if first.seinunit then do;
    naics_1997_valid="0";
    naics_2002_valid="0";
  end;

  naics7="E" || naics;
  naics_1997_invalid="0";
  naics_2002_invalid="0";

  /* Run NAICS through the Multi-format */
  naics_1997_format=put(naics7,$197naic.);
  if substr(naics_1997_format,1,1)="E" then naics_1997_invalid="1";

  naics_2002_format=put(naics7,$102naic.);
  if substr(naics_2002_format,1,1)="E" then naics_2002_invalid="1";

  if naics_1997_invalid="0" then naics_1997_valid="1";
  if naics_2002_invalid="0" then naics_2002_valid="1";

  output temp3;
  if last.seinunit then output temp4;
run;

proc freq data=temp3;
  tables naics_1997_invalid naics_2002_invalid;
run;

proc freq data=temp4;
  tables naics_1997_valid naics_2002_valid;
run;

```

Attach the information showing whether a valid code was found . Clean the industry code if no other code is found . Otherwise set to missing . The valid code will be picked up when filling is done .

```

data temp5;
  merge temp3(in=a) temp4(in=b);
  by sein seinunit;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;

  length naics1997 naics1997_clean naics1997_impute $6;

  /* Set up NAICS 1997 variables */
  naics1997_clean="999999";
  if naics_1997_invalid="0" then naics1997_clean=naics;
  if naics1997~="999999" and naics_1997_invalid="1" and naics_1997_valid="1"
then naics_1997_invalid="6";

  naics1997=naics;
  uniform1=ranuni(4749234);

```

For Industry codes not in the multi-format impute using the first X digits .

```

  candidate1997=0;
  if naics1997~="999999" and naics_1997_invalid="1" and naics_1997_valid="0"
then do;
  candidate1997=1;

```

Check the 5 digit code first .

```

  %indlkup(naics1997,naics1997,5,parms_us_imp_ncs1997_ncs19975,uniform1);
  if naics1997_clean~="999999" then naics_1997_invalid="5";

```

if not found at 5 digit level check 4 digit .

```

  if naics1997_clean="999999" then do;
  %indlkup(naics1997,naics1997,4,parms_us_imp_ncs1997_ncs19974,uniform1);
  end;
  if naics_1997_invalid="1" and naics1997_clean~="999999" then
naics_1997_invalid="4";

```

if not found at 4 digit level check 3 digit .

```

  if naics1997_clean="999999" then do;
  %indlkup(naics1997,naics1997,3,parms_us_imp_ncs1997_ncs19973,uniform1);
  end;
  if naics_1997_invalid="1" and naics1997_clean~="999999" then
naics_1997_invalid="3";
  end;

  length naics2002 naics2002_clean naics2002_impute $6;

  /* Set up NAICS 2002 variables */
  naics2002_clean="999999";
  if naics_2002_invalid="0" then naics2002_clean=naics;
  if naics2002~="999999" and naics_2002_invalid="1" and naics_2002_valid="1"
then naics_2002_invalid="6";

  naics2002=naics;
  uniform1=ranuni(6937541);

```

For Industry codes not in the multi-format impute using the first X digits .

```

candidate2002=0;
if naics2002~="999999" and naics_2002_invalid="1" and naics_2002_valid="0"
then do;
    candidate2002=1;

```

Check the 5 digit code first .

```

%indlkup(naics2002,naics2002,5,parms_us_imp_ncs2002_ncs20025,uniform1);
if naics2002_clean~="999999" then naics_2002_invalid="5";

```

if not found at 5 digit level check 4 digit .

```

if naics2002_clean="999999" then do;
    %indlkup(naics2002,naics2002,4,parms_us_imp_ncs2002_ncs20024,uniform1);
end;
if naics_2002_invalid="1" and naics2002_clean~="999999" then
naics_2002_invalid="4";

```

if not found at 4 digit level check 3 digit .

```

if naics2002_clean="999999" then do;
    %indlkup(naics2002,naics2002,3,parms_us_imp_ncs2002_ncs20023,uniform1);
end;
if naics_2002_invalid="1" and naics2002_clean~="999999" then
naics_2002_invalid="3";
end;
run;

```

DIAGNOSTIC OUTPUT

```

proc contents;
run;

proc freq;
tables merge candidate1997 candidate2002 naics_1997_invalid*naics_1997_valid
naics_2002_invalid*naics_2002_valid;
run;

proc freq data=temp5(where=(naics_1997_invalid~="0"));
tables naics1997_clean*naics_1997_invalid;
run;

proc freq data=temp5(where=(naics_2002_invalid~="0"));
tables naics2002_clean*naics_2002_invalid;
run;

proc print data=temp5(where=(candidate1997=1));
var sein seinunit naics_1997_invalid naics_1997_valid naics1997
naics1997_clean naics1997_impute;
run;

proc print data=temp5(where=(candidate2002=1));
var sein seinunit naics_2002_invalid naics_2002_valid naics2002
naics2002_clean naics2002_impute;
run;

```

Reattach the cleaned NAICS codes back to the original

```

data temp6;
  merge temp1(in=a) temp5(in=b keep=sein seinunit naics naics1997_clean
naics_1997_invalid naics_1997_valid
                                                    naics2002_clean
naics_2002_invalid naics_2002_valid);
  by sein seinunit naics;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;
run;

proc sort data=temp6 out=INTERWRK.naics_clean_02;
  by sein year quarter seinunit;
run;

```

1.5.2.3 library/sasprogs/03_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/sasprogs/03_ecf.sas >--- */
/*----- LEGACY NAME: 03_naics_aux_clean.sas -----*/
/*****
/* REQUIRES-DATA: ecf_stacked_01.sas7bdat */
/*
/* PROVIDES-DATA: naics_aux_clean_03.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Clean and Standardize the NAICS codes";
title2 "Program Location: &curdir./03_naics_aux.sas";

```

Collapse to the set of records with POTENTIALLY valid industry codes .

```

proc sort data=interwrk.ecf_stacked_01(keep=sein year quarter seinunit
naics_aux where=(naics_aux~="999999")) out=temp1;
  by sein seinunit naics_aux year quarter;
run;

```

Collapse to the set of unique industry codes for that SEIN SEINUNIT .

```

proc sort data=temp1(keep=sein seinunit naics_aux) out=temp2 nodupkey;
  by sein seinunit naics_aux;
run;

```

Clean each industry code . Count the number of times a valid code is found .

```

data temp3(drop=naics_aux_1997_valid naics_aux_2002_valid) temp4(keep=sein
seinunit naics_aux_1997_valid naics_aux_2002_valid);
  set temp2;
  by sein seinunit;

  length naics_aux_1997_invalid naics_aux_1997_valid $1 naics_aux_1997_format
$7
         naics_aux_2002_invalid naics_aux_2002_valid $1 naics_aux7
naics_aux_2002_format $7;

  retain naics_aux_1997_valid naics_aux_2002_valid;

/* xxx_valid is whether a valid code is found anywhere during the period */
/* xxx_invalid is whether a valid code is present in the current period */

if first.seinunit then do;
  naics_aux_1997_valid="0";
  naics_aux_2002_valid="0";
end;

naics_aux7="E" || naics_aux;
naics_aux_1997_invalid="0";
naics_aux_2002_invalid="0";

/* Run NAICS through the Multi-format */
naics_aux_1997_format=put(naics_aux7,$197naic.);
if substr(naics_aux_1997_format,1,1)="E" then naics_aux_1997_invalid="1";

naics_aux_2002_format=put(naics_aux7,$102naic.);
if substr(naics_aux_2002_format,1,1)="E" then naics_aux_2002_invalid="1";

if naics_aux_1997_invalid="0" then naics_aux_1997_valid="1";
if naics_aux_2002_invalid="0" then naics_aux_2002_valid="1";

output temp3;
if last.seinunit then output temp4;
run;

```

Diagnostics.

```

proc freq data=temp3;
  tables naics_aux_1997_invalid naics_aux_2002_invalid;
run;

proc freq data=temp4;
  tables naics_aux_1997_valid naics_aux_2002_valid;
run;

```

Attach the information showing whether a valid code was found . Clean the industry code if no other code is found . Otherwise set to missing . The valid code will be picked up when filling is done .

```

data temp5;
  merge temp3(in=a) temp4(in=b);
  by sein seinunit;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;

  length naics_aux1997 naics_aux1997_clean naics1997_impute $6;

  /* Set up NAICS 1997 variables */
  naics_aux1997_clean="999999";
  if naics_aux_1997_invalid="0" then naics_aux1997_clean=naics_aux;
  if naics_aux1997~="999999" and naics_aux_1997_invalid="1" and
naics_aux_1997_valid="1" then naics_aux_1997_invalid="6";

  naics_aux1997=naics_aux;
  uniform1=ranuni(9048234);

```

For Industry codes not in the multi-format impute using the first X digits .

```

  candidate1997=0;
  if naics_aux1997~="999999" and naics_aux_1997_invalid="1" and
naics_aux_1997_valid="0" then do;
    candidate1997=1;

```

Check the 5 digit code first .

```

%indlkup(naics1997,naics_aux1997,5,parms_us_imp_ncs1997_ncs19975,uniform1);
  if naics_aux1997_clean~="999999" then naics_aux_1997_invalid="5";

```

if not found at 5 digit level check 4 digit .

```

  if naics_aux1997_clean="999999" then do;

%indlkup(naics1997,naics_aux1997,4,parms_us_imp_ncs1997_ncs19974,uniform1);
  end;
  if naics_aux_1997_invalid="1" and naics_aux1997_clean~="999999" then
naics_aux_1997_invalid="4";

```

if not found at 4 digit level check 3 digit .

```

  if naics_aux1997_clean="999999" then do;

%indlkup(naics1997,naics_aux1997,3,parms_us_imp_ncs1997_ncs19973,uniform1);
  end;
  if naics_aux_1997_invalid="1" and naics_aux1997_clean~="999999" then
naics_aux_1997_invalid="3";
  end;

```

```

length naics_aux2002 naics_aux2002_clean naics2002_impute $6;

```

```

/* Set up NAICS 2002 variables */
naics_aux2002_clean="999999";
if naics_aux_2002_invalid="0" then naics_aux2002_clean=naics_aux;
if naics_aux2002~="999999" and naics_aux_2002_invalid="1" and
naics_aux_2002_valid="1" then naics_aux_2002_invalid="6";

naics_aux2002=naics_aux;
uniform1=ranuni(6937541);

```

For Industry codes not in the multi-format impute using the first X digits .

```
candidate2002=0;
if naics_aux2002~="999999" and naics_aux_2002_invalid="1" and
naics_aux_2002_valid="0" then do;
    candidate2002=1;
```

Check the 5 digit code first .

```
%indlkup(naics2002,naics_aux2002,5,parms_us_imp_ncs2002_ncs20025,uniform1);
if naics_aux2002_clean~="999999" then naics_aux_2002_invalid="5";
```

if not found at 5 digit level check 4 digit .

```
if naics_aux2002_clean="999999" then do;

%indlkup(naics2002,naics_aux2002,4,parms_us_imp_ncs2002_ncs20024,uniform1);
end;
if naics_aux_2002_invalid="1" and naics_aux2002_clean~="999999" then
naics_aux_2002_invalid="4";
```

if not found at 4 digit level check 3 digit .

```
if naics_aux2002_clean="999999" then do;

%indlkup(naics2002,naics_aux2002,3,parms_us_imp_ncs2002_ncs20023,uniform1);
end;
if naics_aux_2002_invalid="1" and naics_aux2002_clean~="999999" then
naics_aux_2002_invalid="3";
end;
run;
```

DIAGNOSTIC OUTPUT

```

proc contents;
run;

proc freq;
  tables merge candidate1997 candidate2002
  naics_aux_1997_invalid*naics_aux_1997_valid
  naics_aux_2002_invalid*naics_aux_2002_valid;
run;

proc freq data=temp5(where=(naics_aux_1997_invalid~="0"));
  tables naics_aux1997_clean*naics_aux_1997_invalid;
run;

proc freq data=temp5(where=(naics_aux_2002_invalid~="0"));
  tables naics_aux2002_clean*naics_aux_2002_invalid;
run;

proc print data=temp5(where=(candidate1997=1));
  var sein seinunit naics_aux_1997_invalid naics_aux_1997_valid naics_aux1997
  naics_aux1997_clean naics1997_impute;
run;

proc print data=temp5(where=(candidate2002=1));
  var sein seinunit naics_aux_2002_invalid naics_aux_2002_valid naics_aux2002
  naics_aux2002_clean naics2002_impute;
run;

/* Reattach the cleaned NAICS codes back to the original */

data temp6;
  merge temp1(in=a) temp5(in=b keep=sein seinunit naics_aux
  naics_aux1997_clean naics_aux_1997_invalid naics_aux_1997_valid
  naics_aux2002_clean
  naics_aux_2002_invalid naics_aux_2002_valid);
  by sein seinunit naics_aux;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;
run;

proc sort data=temp6 out=INTERWRK.naics_aux_clean_03;
  by sein year quarter seinunit;
run;

```

1.5.2.4 library/sasprogs/04_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/04_ecf.sas >--+ */
/*----- LEGACY NAME: 04_sic_clean.sas -----*/
/*****
/* REQUIRES-DATA: ecf_stacked_01.sas7bdat */
/*
/* PROVIDES-DATA: sic_clean_04.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Clean and Standardize SIC codes";
title2 "Program Location: &curdir./04_sic_clean.sas";
```

Collapse to the set of records with POTENTIALLY valid industry codes .

```
proc sort data=interwrk.ecf_stacked_01(keep=sein year quarter seinunit sic
where=(sic~="9999")) out=temp1;
  by sein seinunit sic year quarter;
run;
```

Collapse to the set of unique industry codes for that SEIN SEINUNIT .

```
proc sort data=temp1(keep=sein seinunit sic) out=temp2 nodupkey;
  by sein seinunit sic;
run;
```

Clean each industry code .

Count the number of times a valid code is found .

```

data temp3(drop=sic_valid) temp4(keep=sein seinunit sic_valid);
  set temp2;
  by sein seinunit;

  length sic_invalid sic_valid $1 sic5 sic_format $5;

  retain sic_valid;

  /* xxx_valid is whether a valid code is found anywhere during the period */
  /* xxx_invalid is whether a valid code is present in the current period */

  if first.seinunit then do;
    sic_valid="0";
  end;

  sic5="D" || sic;
  sic_invalid="0";

  /* Run SIC through the Multi-format */
  sic_format=put(sic5,$1sic.);
  if substr(sic_format,1,1)="D" then sic_invalid="1";

  if sic_invalid="0" then sic_valid="1";

  output temp3;
  if last.seinunit then output temp4;
run;

proc freq data=temp3;
  tables sic_invalid;
run;

proc freq data=temp4;
  tables sic_valid;
run;

```

Attach the information showing whether a valid code was found .

Clean the industry code if no other code is found .

Otherwise set to missing .

The valid code will be picked up when filling is done .

```

data INTERWRK.temp5_04;
  merge temp3(in=a) temp4(in=b);
  by sein seinunit;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;

  length sic sic_clean sic_impute $4;

  /* Set up SIC variables */
  sic_clean="9999";
  if sic_invalid="0" then sic_clean=sic;
  if sic~="9999" and sic_invalid="1" and sic_valid="1" then sic_invalid="6";

  uniform1=ranuni(6859320);

```

For Industry codes not in the multi-format impute using the first X digits .

```

candidate=0;
if sic~="9999" and sic_invalid="1" and sic_valid="0" then do;
  candidate=1;

```

Check the 3 digit code first .

```

%indlkup(sic,sic,3,parms_us_imp_sic_sic3,uniform1);
if sic_clean~="9999" then sic_invalid="3";

```

if not found at 2 digit level check 4 digit .

```

if sic_clean="9999" then do;
  %indlkup(sic,sic,2,parms_us_imp_sic_sic2,uniform1);
end;
if sic_invalid="1" and sic_clean~="9999" then sic_invalid="2";
end;
run;

```

DIAGNOSTIC OUTPUT moved to ecfdiag_04.sas

```

/* Reattach the cleaned SIC codes back to the original */

data temp6;
  merge temp1(in=a) INTERWRK.temp5_04(in=b keep=sein seinunit sic sic_clean
  sic_invalid sic_valid);
  by sein seinunit sic;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;
run;

proc sort data=temp6 out=INTERWRK.sic_clean_04;
  by sein year quarter seinunit;
run;

```

1.5.2.5 library/sasprogs/05_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/05_ecf.sas >--+ */
/*----- LEGACY NAME: 05_merge_ind.sas -----*/
/*****
/* REQUIRES-DATA: ecf_stacked_01.sas7bdat */
/* naics_clean_02.sas7bdat */
/* naics_aux_clean_03.sas7bdat */
/* naics_sic_clean_04.sas7bdat */
/*
/* PROVIDES-DATA: ecf_stacked_clean_05.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Bring together all standardized industry codes with the original data";
title2 "Program Location: &curdir./05_merge_ind.sas";
```

Merge the industry data first .

```
data INTERWRK.ecf_stacked_clean_05;
  merge INTERWRK.ecf_stacked_01(in=a drop=rk1)
        INTERWRK.naics_clean_02(in=b keep=sein year quarter seinunit
naics1997_clean naics_1997_invalid
                                naics2002_clean naics_2002_invalid)
        INTERWRK.naics_aux_clean_03(in=c keep=sein year quarter seinunit
naics_aux1997_clean naics_aux_1997_invalid
                                naics_aux2002_clean
naics_aux_2002_invalid)
        INTERWRK.sic_clean_04(in=d keep=sein year quarter seinunit sic_clean
sic_invalid);
  by sein year quarter seinunit;
```

Create a Merge 4 digit variable .

```

length merge4 $4;
merge4="0000";
if a=1 then substr(merge4,1,1)="1";
if b=1 then substr(merge4,2,1)="1";
if c=1 then substr(merge4,3,1)="1";
if d=1 then substr(merge4,4,1)="1";

/* Clean up the variables */
if naics1997_clean=" " then do;
  naics1997_clean="999999";
  naics_1997_invalid="1";
end;
if naics_aux1997_clean=" " then do;
  naics_aux1997_clean="999999";
  naics_aux_1997_invalid="1";
end;

if naics2002_clean=" " then do;
  naics2002_clean="999999";
  naics_2002_invalid="1";
end;
if naics_aux2002_clean=" " then do;
  naics_aux2002_clean="999999";
  naics_aux_2002_invalid="1";
end;

if sic_clean=" " then do;
  sic_clean="9999";
  sic_invalid="1";
end;

```

Compare the new codes with the originals

```

sic_equal=0;
naics1997_equal=0;
naics2002_equal=0;
naics_aux1997_equal=0;
naics_aux2002_equal=0;

if sic=sic_clean then sic_equal=1;
if naics=naics1997_clean then naics1997_equal=1;
if naics=naics2002_clean then naics2002_equal=1;
if naics_aux=naics_aux1997_clean then naics_aux1997_equal=1;
if naics_aux=naics_aux2002_clean then naics_aux2002_equal=1;
run;

```

Diagnostics moved to ecfdiag_05

1.5.2.6 library/sasprogs/06_ecf.sas

```

/* +---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* +---< Location: /programs/production/dev1/current/ecf >---+ */
/* +---< File: library/sasprogs/06_ecf.sas >---+ */
/*----- LEGACY NAME: 06_num_records.sas -----*/

```

Count the number of records for each SEIN in a given year and quarter. Check whether data is missing for all the records of an SEIN in a given year and quarter

This program creates many of the measures that will be used later to determine the record structure for an SEIN. Unfortunately, the record structure of a multi-unit SEIN is not consistent. An seinunit equal to "00000" is either the only record for a single unit or the master record for a multi-unit. The sub-units of an SEIN should be in the range "00001" to "99999".

However, the data we actual receive contains many variations on the above theme. The variables created here will allow me to create a consistent data structure from the variations on the intended structure that we actually receive from the states.

For the first record of an SEIN I set up initial values for the variables and check whether seinunit="00000". If seinunit~="00000" then I assume that there is no master record and the record represents information about an establishment. This distinction is important when creating the all_miss_XX series of variables. These variables are intended to represent when all of the data for a given SEIN is missing except for the data in the master record. I will handle the data in the master record separately in the next program.

```

/*****
/* REQUIRES-DATA: ecf_stacked_clean_05.sas7bdat */
/* PROVIDES-DATA: count_data_06.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Count the number of records for each SEIN YEAR QUARTER";
title2 "Program Location: &curdir./06_num_records.sas";

```

Remove duplicates (state specific dups only) from the data .

THIS PORTION OF THE CODE REMOVED ON 1/19/2004 by KEVIN MCKINNEY

Process the records that are left .

```

data temp_all(drop=num_records all_miss_pay all_miss_emp1 all_miss_emp2
all_miss_emp3
              all_miss_sic all_miss_county all_miss_owner_code
              all_miss_naics1997 all_miss_naics2002 all_miss_naics_aux1997
all_miss_naics_aux2002)
  temp_sein(keep=sein year quarter num_records all_miss_pay all_miss_emp1
all_miss_emp2 all_miss_emp3
            all_miss_sic all_miss_county all_miss_owner_code
            all_miss_naics1997 all_miss_naics2002 all_miss_naics_aux1997
all_miss_naics_aux2002);
  set INTERWRK.ecf_stacked_clean_05(drop=duprec ein_num ein_char2 county_temp
ein sic naics naics_aux sic_equal naics1997_equal naics2002_equal
naics_aux1997_equal naics_aux2002_equal
merge4 rename=(temp_ein=ein sic_clean=sic
naics1997_clean=naics1997
naics2002_clean=naics2002
naics_aux1997_clean=naics_aux1997
naics_aux2002_clean=naics_aux2002));
  by sein year quarter;

  retain NUM_RECORDS all_miss_pay all_miss_emp1 all_miss_emp2 all_miss_emp3
all_miss_sic all_miss_county all_miss_owner_code
all_miss_naics1997 all_miss_naics2002 all_miss_naics_aux1997
all_miss_naics_aux2002;

  label NUM_RECORDS='Number of Records by SEIN in ES202';

```

First-quarter processing.

```

if first.quarter then do;
  NUM_RECORDS=0;
  all_miss_pay=1;
  all_miss_emp1=1;
  all_miss_emp2=1;
  all_miss_emp3=1;
  all_miss_sic=1;
  all_miss_naics1997=1;
  all_miss_naics2002=1;
  all_miss_naics_aux1997=1;
  all_miss_naics_aux2002=1;
  all_miss_county=1;
  all_miss_owner_code=1;
  if seinunit~="00000" then do;
    if total_wages>0 then all_miss_pay=0;
    if empl_month1>0 then all_miss_emp1=0;
    if empl_month2>0 then all_miss_emp2=0;
    if empl_month3>0 then all_miss_emp3=0;
    if sic~="9999" then all_miss_sic=0;
    if naics1997~="999999" then all_miss_naics1997=0;
    if naics2002~="999999" then all_miss_naics2002=0;
    if naics_aux1997~="999999" then all_miss_naics_aux1997=0;
    if naics_aux2002~="999999" then all_miss_naics_aux2002=0;
    if county~="ZZZ" then all_miss_county=0;
    if owner_code~="9" then all_miss_owner_code=0;
  end;
end;

```

Increment counter.

```
NUM_RECORDS=NUM_RECORDS+1;
```

Set missing flags.

```

if first.quarter and last.quarter and total_wages>0 then all_miss_pay=0;
if first.quarter and last.quarter and empl_month1>0 then all_miss_emp1=0;
if first.quarter and last.quarter and empl_month2>0 then all_miss_emp2=0;
if first.quarter and last.quarter and empl_month3>0 then all_miss_emp3=0;
if first.quarter and last.quarter and sic~="9999" then all_miss_sic=0;
if first.quarter and last.quarter and naics1997~="999999" then
all_miss_naics1997=0;
  if first.quarter and last.quarter and naics2002~="999999" then
all_miss_naics2002=0;
  if first.quarter and last.quarter and naics_aux1997~="999999" then
all_miss_naics_aux1997=0;
  if first.quarter and last.quarter and naics_aux2002~="999999" then
all_miss_naics_aux2002=0;
  if first.quarter and last.quarter and county~="ZZZ" then all_miss_county=0;
  if first.quarter and last.quarter and owner_code~="9" then
all_miss_owner_code=0;

if total_wages>0 and num_records>1 then all_miss_pay=0;
if empl_month1>0 and num_records>1 then all_miss_emp1=0;
if empl_month2>0 and num_records>1 then all_miss_emp2=0;
if empl_month3>0 and num_records>1 then all_miss_emp3=0;
if sic~="9999" and num_records>1 then all_miss_sic=0;
if naics1997~="999999" and num_records>1 then all_miss_naics1997=0;
if naics2002~="999999" and num_records>1 then all_miss_naics2002=0;
if naics_aux1997~="999999" and num_records>1 then all_miss_naics_aux1997=0;
if naics_aux2002~="999999" and num_records>1 then all_miss_naics_aux2002=0;
if county~="ZZZ" and num_records>1 then all_miss_county=0;
if owner_code~="9" and num_records>1 then all_miss_owner_code=0;

```

Break out yearly and quarterly files.

```

output temp_all;
if last.quarter then do;
    output temp_sein;
end;
run;

```

Diagnostics.

```

proc freq data=temp_sein;
title3 "SEIN YEAR QUARTER records";
tables num_records /v5fmt;
run;

data INTERWRK.count_data_06;
merge temp_all temp_sein;
by sein year quarter;
run;

```

Diagnostics moved to ecfdiag_06.sas

1.5.2.7 library/sasprogs/07_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/sasprogs/07_ecf.sas >---+ */
/*----- LEGACY NAME: 07_rm_master.sas -----*/

```

Eliminate the master record while preserving any information in the subunit records. Create multi-unit indicator and establishment counts.

This program is extremely important since it creates the record structure that will be used from this point forward. If this process is not done carefully then important information about the SEIN may be destroyed.

I break the processing tasks up into three distinct groups.

1. SEIN's with only one record
2. The first record for SEIN's with more than one record
3. The second record and beyond for SEIN's with more than one record

Detailed Explanation SEIN's with only one record must be single units. Therefore I set the num_estabs=1 and initialize all other variables. If all_miss_XX=1 then I set no_XX=1 (the subunit is the master record in this case). There is also no need to allocate master record information to the subunits.

For the other cases, initialize all variables and if the master record is present then store any information in the master record in the master_XX variables. Calculate the number of establishments if there is a master record present by subtracting one from the number of records calculated previously. If the number of records is equal to two and one of the records is a master record then this situation is inconsistent with the defined record structure. The number of establishments is set equal to one in this case. This situation likely occurs when a multi-unit shrinks and only one unit is left. The master record is not removed immediately since it is unclear whether the SEIN will remain a single unit or return to multi-unit status. If there is information in the master record and there is no information in the subunits then set the impute_XX flag=1. If there is no information in the master and the subunits then set the no_XX flag=1. Finally, if there is no master record then calculate the number of establishments as equal to the number of records. If the number of establishments is greater than one then set multi_unit=1.

The second record and beyond are output without modification unless the impute_XX flag=1. In this case, information from the master record is allocated to the subunits. Since I have no additional information

at this point about the structure of the SEINUNITS I allocate any payroll or employment equally across the subunits. Further along in the sequence I use additional information available in years and quarters when the subunits report payroll and employment to improve the allocation.

```

/*****
/* REQUIRES-DATA: count_data_06.sas7bdat */
/* PROVIDES-DATA: no_master_07.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Remove the Master Record from /data/master/ui/&state./YYYY";
title2 "Program Location: &curdir./07_rm_master.sas";

data INTERWRK.no_master_07(drop=master_wage--master_emp3 master_sic
master_naics1997 master_naics2002 master_naics_aux1997 master_naics_aux2002
master_county master_owner_code
all_miss_pay--all_miss_emp3 all_miss_sic all_miss_county all_miss_owner_code
all_miss_naics1997 all_miss_naics2002
all_miss_naics_aux1997 all_miss_naics_aux2002);
set INTERWRK.count_data_06();
by sein year quarter;

retain impute_wage impute_emp1 impute_emp2 impute_emp3
master_wage master_emp1 master_emp2 master_emp3
no_wages no_emp1 no_emp2 no_emp3 NUM_ESTABS multi_unit

impute_sic impute_county impute_owner_code
master_sic master_county master_owner_code
no_sic no_county no_owner_code

impute_naics1997 impute_naics2002 impute_naics_aux1997
impute_naics_aux2002
master_naics1997 master_naics2002 master_naics_aux1997
master_naics_aux2002
no_naics1997 no_naics2002 no_naics_aux1997 no_naics_aux2002

master_empl_month1_flg master_empl_month2_flg master_empl_month3_flg
master_total_wages_flg master_multi_unit_code;

length master_sic $4 master_naics1997 master_naics2002 master_naics_aux1997
master_naics_aux2002 $6 master_county $3
master_owner_code master_empl_month1_flg master_empl_month2_flg
master_empl_month3_flg master_total_wages_flg master_multi_unit_code $1;

```

Process for all records

```

if seinunit="00000" then seinunit_type=0;
else seinunit_type=1;

```

If there is only one record then handle immediately

```

if first.quarter and last.quarter then do;
  NUM_ESTABS=1;
  multi_unit=0;
  impute_wage=0;
  impute_emp1=0;
  impute_emp2=0;
  impute_emp3=0;
  impute_sic=0;
  impute_naics1997=0;
  impute_naics2002=0;
  impute_naics_aux1997=0;
  impute_naics_aux2002=0;
  impute_county=0;
  impute_owner_code=0;
  no_wages=0;
  no_emp1=0;
  no_emp2=0;
  no_emp3=0;
  no_sic=0;
  no_naics1997=0;
  no_naics2002=0;
  no_naics_aux1997=0;
  no_naics_aux2002=0;
  no_county=0;
  no_owner_code=0;
  master_wage=0;
  master_emp1=0;
  master_emp2=0;
  master_emp3=0;
  master_sic="9999";
  master_naics1997="999999";
  master_naics2002="999999";
  master_naics_aux1997="999999";
  master_naics_aux2002="999999";
  master_county="ZZZ";
  master_owner_code="9";
  /* We want to keep the master for these variables */
  master_multi_unit_code=" ";
  master_empl_month1_flg=" ";
  master_empl_month2_flg=" ";
  master_empl_month3_flg=" ";
  master_total_wages_flg=" ";
  if all_miss_pay=1 then no_wages=1;
  if all_miss_emp1=1 then no_emp1=1;
  if all_miss_emp2=1 then no_emp2=1;
  if all_miss_emp3=1 then no_emp3=1;
  if all_miss_sic=1 then no_sic=1;
  if all_miss_naics1997=1 then no_naics1997=1;
  if all_miss_naics2002=1 then no_naics2002=1;
  if all_miss_naics_aux1997=1 then no_naics_aux1997=1;
  if all_miss_naics_aux2002=1 then no_naics_aux2002=1;
  if all_miss_county=1 then no_county=1;
  if all_miss_owner_code=1 then no_owner_code=1;
  output;
  *put sein= year= quarter= seinunit= master_emp1= empl_month1=
impute_wage=;
end;

```

Process the first quarter for multi-record firms If the first record is not a master (seinunit=00000) then output.

```

if first.quarter and not last.quarter then do;
NUM_ESTABS=0;
multi_unit=0;
impute_wage=0;
impute_emp1=0;
impute_emp2=0;
impute_emp3=0;
impute_sic=0;
impute_naics1997=0;
impute_naics2002=0;
impute_naics_aux1997=0;
impute_naics_aux2002=0;
impute_county=0;
impute_owner_code=0;
no_wages=0;
no_emp1=0;
no_emp2=0;
no_emp3=0;
no_sic=0;
no_naics1997=0;
no_naics2002=0;
no_naics_aux1997=0;
no_naics_aux2002=0;
no_county=0;
no_owner_code=0;
master_wage=0;
master_emp1=0;
master_emp2=0;
master_emp3=0;
master_sic="9999";
master_naics1997="999999";
master_naics2002="999999";
master_naics_aux1997="999999";
master_naics_aux2002="999999";
master_county="ZZZ";
master_owner_code="9";

```

We want to keep the master for these variables

```

master_multi_unit_code=" ";
master_empl_month1_flg=" ";
master_empl_month2_flg=" ";
master_empl_month3_flg=" ";
master_total_wages_flg=" ";

```

If master record present then store master wages and calc number of estabs

```

if seinunit="00000" then do;
  master_wage=total_wages;
  master_emp1=empl_month1;
  master_emp2=empl_month2;
  master_emp3=empl_month3;
  master_sic=sic;
  master_naics1997=naics1997;
  master_naics2002=naics2002;
  master_naics_aux1997=naics_aux1997;
  master_naics_aux2002=naics_aux2002;
  master_county=county;
  master_owner_code=owner_code;
  master_multi_unit_code=multi_unit_code;
  master_empl_month1_flg=empl_month1_flg;
  master_empl_month2_flg=empl_month1_flg;
  master_empl_month3_flg=empl_month1_flg;
  master_total_wages_flg=total_wages_flg;
  if num_records<3 then num_estabs=1;
  if num_records>2 then num_estabs=num_records-1;
  if num_estabs>1 then multi_unit=1;
end;

```

Set impute flag if all units except master have no data

```

  if master_wage>0 and all_miss_pay=1 then impute_wage=1;
  if master_emp1>0 and all_miss_emp1=1 then impute_emp1=1;
  if master_emp2>0 and all_miss_emp2=1 then impute_emp2=1;
  if master_emp3>0 and all_miss_emp3=1 then impute_emp3=1;
  if master_sic~="9999" and all_miss_sic=1 then impute_sic=1;
  if master_naics1997~="999999" and all_miss_naics1997=1 then
impute_naics1997=1;
  if master_naics2002~="999999" and all_miss_naics2002=1 then
impute_naics2002=1;
  if master_naics_aux1997~="999999" and all_miss_naics_aux1997=1 then
impute_naics_aux1997=1;
  if master_naics_aux2002~="999999" and all_miss_naics_aux2002=1 then
impute_naics_aux2002=1;
  if master_county~="ZZZ" and all_miss_county=1 then impute_county=1;
  if master_owner_code~="9" and all_miss_owner_code=1 then
impute_owner_code=1;

```

If no wage data in master and subunits then set no_wages flag

```

  if not(total_wages>0) and all_miss_pay=1 then no_wages=1;
  if not(empl_month1>0) and all_miss_emp1=1 then no_emp1=1;
  if not(empl_month2>0) and all_miss_emp2=1 then no_emp2=1;
  if not(empl_month3>0) and all_miss_emp3=1 then no_emp3=1;
  if sic="9999" and all_miss_sic=1 then no_sic=1;
  if naics1997="999999" and all_miss_naics1997=1 then no_naics1997=1;
  if naics2002="999999" and all_miss_naics2002=1 then no_naics2002=1;
  if naics_aux1997="999999" and all_miss_naics_aux1997=1 then
no_naics_aux1997=1;
  if naics_aux2002="999999" and all_miss_naics_aux2002=1 then
no_naics_aux2002=1;
  if county="ZZZ" and all_miss_county=1 then no_county=1;
  if owner_code="9" and all_miss_owner_code=1 then no_owner_code=1;

```

If no master record then calculate num_estabs and output data

```

if seinunit~="00000" then do;
  if num_records<3 then num_estabs=2;
  if num_records>2 then num_estabs=num_records;
  if num_estabs>1 then multi_unit=1;
  output;
end;
end;

```

Process the second to the last record of multi record firms

```

if not first.quarter then do;
  if impute_wage=1 then do;
    total_wages=master_wage/(num_records-1);
  end;
  if impute_emp1=1 then do;
    empl_month1=master_emp1/(num_records-1);
  end;
  if impute_emp2=1 then do;
    empl_month2=master_emp2/(num_records-1);
  end;
  if impute_emp3=1 then do;
    empl_month3=master_emp3/(num_records-1);
  end;
  if impute_sic=1 then do;
    sic=master_sic;
  end;
  if impute_naics1997=1 then do;
    naics1997=master_naics1997;
  end;
  if impute_naics2002=1 then do;
    naics2002=master_naics2002;
  end;
  if impute_naics_aux1997=1 then do;
    naics_aux1997=master_naics_aux1997;
  end;
  if impute_naics_aux2002=1 then do;
    naics_aux2002=master_naics_aux2002;
  end;
  if impute_county=1 then do;
    county=master_county;
  end;
  if impute_owner_code=1 then do;
    owner_code=master_owner_code;
  end;
  output;
  *put sein= year= quarter= seinunit= master_emp1= empl_month1=
  impute_wage=;
  end;
  if impute_emp1=1 then do;
    *put sein= year= quarter= seinunit= master_wage= total_wages=;
    *put sein= year= quarter= seinunit= master_emp1= empl_month1=;
  end;
run;

```

Diagnostics moved to ecfdiag_07.sas

1.5.2.8 library/sasprogs/08_ecf.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* +--< Location: /programs/production/dev1/current/ecf >--- */
/* +--< File: library/sasprogs/08_ecf.sas >--- */
/*----- LEGACY NAME: 08_sein_totals.sas -----*/

```

Create the SEIN level totals, indicators for the appearance of wage and employment data, and merge the result with the UI SEIN level data

In the previous program I created an establishment level dataset. Now I am able to sum over the establishment level records and create SEIN totals. I also want to check and see if the SEIN was ever active during the time period that we observe the firm. This information will be used later when I improve my allocation of master record information to the subunits.

Next, I merge UI SEIN totals with SEIN totals from the 202 and create indicators for each SEIN YEAR QUARTER showing what data is available.

Finally, I merge the combined UI and 202 data with a list of the SEIN's that appear on the 202. This allows me to create an indicator showing whether an SEIN ever appeared on the 202. The 202, not the UI contains information on the structure of employment at the subunits, thus it is important to know whether that information may be available.

```

/*****
/* REQUIRES-DATA: no_master_07.sas7bdat */
/* UI sein year quarter totals from /data/master/ECF/uitotals */
/* name is of the form sein_employment_ss.sas7bdat */
/* PROVIDES-DATA: sein_totals_08.sas7bdat, sein_list_ui_08.sas7bdat */
/* sein_list_202_08.sas7bdat, seinunit_202_UI_08.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Create an SEIN level dataset from /data/master/ui/&state./YYYY";
title2 "Program Location: &curdir./08_sein_totals.sas";

```

These ever_XXX variables are only valid for multiunits with measure reported at the unit level .

```

data INTERWRK.sein_totals_08(keep=sein year quarter sein_emp1 sein_emp2
sein_emp3 sein_wages)
INTERWRK.sein_list_202_08(keep=sein ever_multi ever_wages ever_emp1
ever_emp2 ever_emp3
multi_first_year multi_first_quarter);
set INTERWRK.no_master_07;
by sein year quarter;

retain sein_emp1 sein_emp2 sein_emp3 sein_wages multi_first_year
multi_first_quarter
ever_multi ever_wages ever_emp1 ever_emp2 ever_emp3;

```

Initializations.

```

if first.sein then do;
ever_multi=0;
ever_wages=0;
ever_emp1=0;
ever_emp2=0;
ever_emp3=0;
multi_first_year=.;
multi_first_quarter=.;
end;

if first.quarter then do;
sein_emp1=0;
sein_emp2=0;
sein_emp3=0;
sein_wages=0;
end;

```

Summing up.

```

if empl_month1>0 then sein_emp1=sein_emp1+empl_month1;
if empl_month2>0 then sein_emp2=sein_emp2+empl_month2;
if empl_month3>0 then sein_emp3=sein_emp3+empl_month3;
if total_wages>0 then sein_wages=sein_wages+total_wages;

if ever_multi=0 and multi_unit=1 then do;
    ever_multi=1;
    multi_first_year=year;
    multi_first_quarter=quarter;
end;
if multi_unit=1 then do;
    if no_wages=0 and impute_wage=0 then ever_wages=1;
    if no_emp1=0 and impute_emp1=0 then ever_emp1=1;
    if no_emp2=0 and impute_emp2=0 then ever_emp2=1;
    if no_emp3=0 and impute_emp3=0 then ever_emp3=1;
end;

*put sein year quarter empl_month1= sein_emp1=;
if last.quarter then do;
    output INTERWRK.sein_totals_08;
end;

if last.sein then do;
    output INTERWRK.sein_list_202_08;
end;
run;

```

Diagnostics moved to ecfdiag_08.sas

Merge the ES202 with the UI

```

data
  step0
  INTERWRK.sein_list_UI_08(keep=sein);
merge
  /* NEWNAME:
  * INPUTS.ehf_&state._sein_employment
  */
  INPUTS.ehf_&state._sein_employment
  (in=a
   rename=(m=seinsize_m b=seinsize_b e=seinsize_e w1=payroll)
   where=(yr_qtr>="&start_year.:&start_quarter." and
yr_qtr<="&end_year.:&end_quarter.")
  )
  INTERWRK.sein_totals_08(in=b);
  by sein year quarter;

  if a=1 and b=1 then source=3;
  if a=1 and b=0 then source=1;
  if a=0 and b=1 then source=2;

  label source="1=UI only,2=202 only,3=both";

  retain counter_ui;

  if first.sein then do;
    counter_ui=0;
  end;

  in_UI=0; in_202=0;

  if source=3 then do;
    in_UI=1;
    in_202=1;
  end;
  if source=1 then in_UI=1;
  if source=2 then in_202=1;

  if in_UI=1 then counter_ui+1;

  if last.sein then do;
    if counter_ui>0 then output INTERWRK.sein_list_UI_08;
  end;
  output step0;
run;

```

START QA PREP

```

proc freq data=step0;
  title3 "Merge of UI SEIN YEAR QUARTER data with 202 SEIN YEAR QUARTER data";
  tables source in_UI*in_202;
  tables year*source;
run;

```

END QA PREP

Merge the ES202/UI SEIN YEAR QUARTER data with the ES202 SEIN list

```

data step1;
  merge
    step0(in=a)
    INTERWRK.sein_list_202_08(in=b);
  by sein;

  if a=1 and b=1 then _merge=3;
  if a=1 and b=0 then _merge=1;
  if a=0 and b=1 then _merge=2;

  ever_202=0;
  if _merge=3 then ever_202=1;
run;

```

START QA PREP

```

proc contents;
title3 "ES202 and UI SEIN YEAR Quarter data";
run;

proc freq;
tables _merge ever_202;
run;

```

END QA PREP

/* Merge the UI SEIN list with the UI/202 SEIN YEAR QUARTER data */

```

data step2;
  merge
    step1(in=a drop=_merge)
    INTERWRK.sein_list_UI_08(in=b);
  by sein;

  if a=1 and b=1 then _merge=3;
  if a=1 and b=0 then _merge=1;
  if a=0 and b=1 then _merge=2;

  ever_UI=0;
  if _merge=3 then ever_UI=1;
run;

```

START QA PREP

```

proc contents;
title3 "ES202 and UI SEIN YEAR Quarter data";
run;

proc freq;
tables _merge ever_UI ever_UI*ever_202;
run;

```

END QA PREP

```

/* Bring everything together by merging the 202/UI SEIN YEAR QUARTER */
/* data with the 202 SEIN YEAR QUARTER SEINUNIT data */

data INTERWRK.seinunit_202_UI_08;

merge step2(in=a drop=_merge) INTERWRK.no_master_07(in=b);
  by sein year quarter;

if a=1 and b=1 then _merge=3;
if a=1 and b=0 then _merge=1;
if a=0 and b=1 then _merge=2;

if seinunit=" " then seinunit="00000";
if num_estabs=. then num_estabs=1;
if multi_unit=. then multi_unit=0;
if county=" " then county="ZZZ";
if sic=" " then sic="9999";
if naics1997=" " then naics1997="999999";
if naics2002=" " then naics2002="999999";
if naics_aux1997=" " then naics_aux1997="999999";
if naics_aux2002=" " then naics_aux2002="999999";
if owner_code=" " then owner_code="9";
if ever_emp1=. then ever_emp1=0;
if ever_emp2=. then ever_emp2=0;
if ever_emp3=. then ever_emp3=0;
if ever_wages=. then ever_wages=0;
if ever_multi=. then ever_multi=0;
if sic_invalid=" " then sic_invalid="1";
if naics1997_invalid=" " then naics1997_invalid="1";
if naics2002_invalid=" " then naics2002_invalid="1";
if naics_aux1997_invalid=" " then naics_aux1997_invalid="1";
if naics_aux2002_invalid=" " then naics_aux2002_invalid="1";
if ein_bad=. then ein_bad=1;
if valid_ein=. then valid_ein=0;
if ein_defect=. then ein_defect=1;
if seinunit_bad=. then seinunit_bad=0;
if seinunit_type=. then seinunit_type=0;
/* Clean Up Payroll and Employment */
if not empl_month1>0 then empl_month1=0;
if not empl_month2>0 then empl_month2=0;
if not empl_month3>0 then empl_month3=0;
if not total_wages>0 then total_wages=0;
if not sein_emp1>0 then sein_emp1=0;
if not sein_emp2>0 then sein_emp2=0;
if not sein_emp3>0 then sein_emp3=0;
if not sein_wages>0 then sein_wages=0;
if not seinsize_m>0 then seinsize_m=0;
if not seinsize_b>0 then seinsize_b=0;
if not seinsize_e>0 then seinsize_e=0;
if not payroll>0 then payroll=0;
/* Create Continuous Time YEAR QUARTER variable */
if yr_qtr=" " then yr_qtr= put(year,4.) || ":" || put(quarter,1.);
run;

```

Diagnostics moved to ecfdiag_08.sas

```

proc sort data=INTERWRK.seinunit_202_UI_08;
  by sein year quarter seinunit;
run;

```

1.5.2.9 library/sasprogs/09_ecf.sas

```
/* +---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* +---< Location: /programs/production/dev1/current/ecf >---+ */
/* +---< File: library/sasprogs/09_ecf.sas >---+ */
/*----- LEGACY NAME: 09_best_vars.sas -----*/
```

This program creates a set of best_XXX variables.

The best_(wages and employment) variables incorporate information from both the UI and the ES202. This analysis is done at the SEINUNIT level.

I examine each record to find out what type of information is available on the ES202. I classify a record into one of four possible categories using the variable info_202.

I look at single-unit and multi-unit records separately. For single-units it is relatively easy to fill wages and employment with information from the UI. For multi-units I do a naive allocation that is improved upon for some firms in 07_special_handle.sas.

UI data is used in the following situations

- ES202 employment is missing, but ES202 payroll is reported, then UI employment is used (if ES202 employment is zero, then UI employment is NOT used since this may be a correct report and I do not have enough information to determine if it is incorrect).
- ES202 payroll is zero and ES202 employment is positive then UI payroll is used.
- If ES202 payroll and employment are zero then UI payroll and employment are used.
- If both ES202 payroll and employment are missing then UI payroll and employment are used.

```
/*-----*/
/* REQUIRES-DATA: seinunit_202_UI_08.sas7bdat */
/* PROVIDES-DATA: best_vars_09.sas7bdat */
/*-----*/

* %include "runtime_ecf.sas";

title1 "Create an SEIN level dataset from /data/master/ui/&state./YYYY";
title2 "Program Location: &curdir./09_best_vars.sas";

data INTERWRK.best_vars_09;

    set INTERWRK.seinunit_202_UI_08(drop=_merge);
```

Initialize Best Wages variables .

```
best_wages=. ;
best_emp1=. ;
best_emp2=. ;
best_emp3=. ;
best_flag=0;
```

Create a set of UI variables that can be used . to fill missing ES202 employment .

```

emp1_UI=0;
emp2_UI=0;
emp3_UI=0;
if in_UI=1 then do;
  emp1_UI=seinsize_b;
  if not(emp1_UI>0) then emp1_UI=seinsize_e;
  if not(emp1_UI>0) then emp1_UI=seinsize_m;

  emp2_UI=seinsize_b;
  if not(emp2_UI>0) then emp2_UI=seinsize_e;
  if not(emp2_UI>0) then emp2_UI=seinsize_m;

  emp3_UI=seinsize_e;
  if not(emp3_UI>0) then emp3_UI=seinsize_b;
  if not(emp3_UI>0) then emp3_UI=seinsize_m;
end;

```

Find out the information on the 202 .

- info_202=0 no wages and no employment
- info_202=1 wages and no employment
- info_202=2 no wages and employment
- info_202=3 wages and employment

```

noemp_202=not(sein_emp1>0) and not(sein_emp2>0) and not(sein_emp3>0);

info_202=0;
if in_202=1 then do;
  if sein_wages>0 and noemp_202=1 then info_202=1;
  if not(sein_wages>0) and noemp_202=0 then info_202=2;
  if sein_wages>0 and noemp_202=0 then info_202=3;
end;

```

Create the best_wages variable, for single establishments.

```

if num_estabs<2 then do;
  if in_202=1 and in_UI=1 then do;
    /* Section 1 */
    if info_202=1 then do;
      best_wages=total_wages;
      best_emp1=emp1_UI;
      best_emp2=emp2_UI;
      best_emp3=emp3_UI;
      best_flag=1;
    end;
    /* Section 2 */
    else if info_202=2 then do;
      best_wages=payroll;
      best_emp1=empl_month1;
      best_emp2=empl_month2;
      best_emp3=empl_month3;
      best_flag=2;
    end;
    /* Section 3 */
    else if info_202=3 then do;
      best_wages=total_wages;
      best_emp1=empl_month1;
      best_emp2=empl_month2;
      best_emp3=empl_month3;
      best_flag=3;
    end;
    /* Section 4 */
    else if info_202=0 then do;
      best_wages=payroll;
      best_emp1=emp1_UI;
      best_emp2=emp2_UI;
      best_emp3=emp3_UI;
      best_flag=4;
    end;
  end;
  /* Section 5 */
  else if in_202=1 and in_UI=0 then do;
    best_wages=total_wages;
    best_emp1=empl_month1;
    best_emp2=empl_month2;
    best_emp3=empl_month3;
    best_flag=5;
  end;
  /* Section 6 */
  else if in_202=0 and in_UI=1 then do;
    best_wages=payroll;
    best_emp1=emp1_UI;
    best_emp2=emp2_UI;
    best_emp3=emp3_UI;
    best_flag=6;
  end;
end;

```

Create the best_wages variable, for multi-establishments.

```

if num_estabs>1 then do;
  if in_202=1 and in_UI=1 then do;
    /* Section 7 */
    if info_202=1 then do;
      best_wages=total_wages;
      best_emp1=emp1_UI*(total_wages/sein_wages);
      best_emp2=emp2_UI*(total_wages/sein_wages);
      best_emp3=emp3_UI*(total_wages/sein_wages);
      best_flag=7;
    end;
    /* Section 8 */
    else if info_202=2 then do;

best_wages=payroll*(empl_month1+empl_month2+empl_month3)/(sein_emp1+sein_emp2+se
in_emp3);

      best_emp1=empl_month1;
      best_emp2=empl_month2;
      best_emp3=empl_month3;
      best_flag=8;
    end;
    /* Section 9 */
    else if info_202=3 then do;
      best_wages=total_wages;
      best_emp1=empl_month1;
      best_emp2=empl_month2;
      best_emp3=empl_month3;
      best_flag=9;
    end;
    /* Section 10 */
    else if info_202=0 then do;
      best_wages=payroll/num_estabs;
      best_emp1=emp1_UI/num_estabs;
      best_emp2=emp2_UI/num_estabs;
      best_emp3=emp3_UI/num_estabs;
      best_flag=10;
    end;
    /* Section 11 */
    else if in_202=1 and in_UI=0 then do;
      best_wages=total_wages;
      best_emp1=empl_month1;
      best_emp2=empl_month2;
      best_emp3=empl_month3;
      best_flag=11;
    end;
    /* Section 12 */
    /* This category is not possible */
    /* If you are in the UI only then must be one record */
  end;
end;
run;

```

Diagnostics moved to ecfdiag_09.sas

1.5.2.10 library/sasprogs/10_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/sasprogs/10_ecf.sas >---+ */
/*----- LEGACY NAME: 10_select_records.sas -----*/

```

Select the records that have enough information available to enable subunit structure to be imputed.

There are four main classes of records that may benefit from using information on subunit structure in other quarters.

1. Records that appear only in the UI in a given year and quarter but appear previously in the 202 as a multi unit.
2. Records that appear only in the 202 and master record data was allocated to the subunits.
3. Records that appear in both the UI and the 202 but have no information in the 202 and are a multi unit.
4. Records that appear in both the UI and the 202 and master record data was allocated to the subunits.

However, not every case that meets the conditions above will have enough prior information to allow an impute to take place. The firm must have been a multi-unit prior to the date that a record met one of the conditions above. The firm must also have valid data in another quarter than can be used for the impute. Only about half of the candidate records contain valid data in Illinois.

All SEINUNIT records selected for imputation are consolidated to SEIN YEAR QUARTER records.

The complete firm history for any SEIN that meets the criteria for special handling is kept in a separate dataset and is indexed for easy access (special_handle_history_10.sas7bdat).

```
/******  
/* REQUIRES-DATA: best_vars_09.sas */  
/* PROVIDES-DATA: alldata_10.sas7bdat, special_handle_history_10.sas7bdat */  
/******  
  
title1 "Calculate totals for the weights and select records for structure  
impute";  
title2 "Program Location: &curdir./10_select_records.sas";
```

Code for control totals removed in 3.1.15, moved to EHF.

Select Records for structure impute

```

data INTERWRK.alldata_10
  (keep=
    sein year quarter seinunit yr_qtr NUM_ESTABS SIC sic_invalid owner_code
    wages_202 emp1_202 emp2_202 emp3_202 auxiliary_code naics1997 naics2002
    naics_aux1997 naics_aux2002 empl_month1_flg empl_month2_flg
empl_month3_flg
    total_wages_flg master_empl_month1_flg master_empl_month2_flg
    master_empl_month3_flg master_total_wages_flg master_multi_unit_code
    multi_unit_code naics_1997_invalid naics_2002_invalid
    naics_aux_1997_invalid naics_aux_2002_invalid best_wages best_emp1
    best_emp2 best_emp3 best_flag county ein ein_bad ein_defect valid_ein
    empl_month1 empl_month2 empl_month3 total_wages ever_202 ever_UI
ever_emp1
    ever_emp2 ever_emp3 ever_multi ever_wages in_202 in_UI
multi_first_quarter
    multi_first_year multi_unit no_wages no_emp1 no_emp2 no_emp3 no_county
    no_naics1997 no_naics2002 no_naics_aux1997 no_naics_aux2002 no_sic
    seinunit_bad seinunit_type source special_handle data_avail impute_wage
    impute_emp1 impute_emp2 impute_emp3 sein_wages sein_emp1 sein_emp2
    sein_emp3 payroll seinsize_m seinsize_b seinsize_e emp1_UI emp2_UI
emp3_UI
    wages_UI impute_data no_data no_get_data qtime_master qtime_first
  )
  INTERWRK.special_handle_list_10
  (keep=sein year quarter qtime_master in_202 in_UI source special_handle
    wages_202 emp1_202 emp2_202 emp3_202 emp1_UI emp2_UI emp3_UI
    wages_UI get_wages get_emp1 get_emp2 get_emp3);

set INTERWRK.best_vars_09();
  by sein year quarter;

*****;

```

Some records will require additional handling before they can be included in the master file.

- UI ONLY - If ever a multi-unit according to the 202 then look for firm structure in another close quarter
- ES202 ONLY - If ES202 master has employment and all subunits do not, then look in another close quarter for structure
- UI and ES202 - If ES202 master has employment or ES202 master does not have employment then output. UI employment can substitute for ES202 master information.

```

*****;

retain special_handle impute_data no_data qtime_master qtime_first
  get_wages get_emp1 get_emp2 get_emp3 no_get_data data_avail;

```

Store the ES202 and UI wage and payroll totals

```

wages_202=sein_wages;
emp1_202=sein_emp1;
emp2_202=sein_emp2;
emp3_202=sein_emp3;

```

UI employment totals already created on 05.

```
wages_UI=payroll;
if first.quarter then do;
```

Was any data allocated from the master to subunits?

```
impute_data=impute_wage=1 or impute_emp1=1 or impute_emp2=1 or
impute_emp3=1;
```

Was all 202 data missing in the master and all subunits?

```
no_data=no_wages=1 and no_emp1=1 and no_emp2=1 and no_emp3=1;
```

Set the current quarter time and the time first appear as multi

```
%qtime(year,quarter) /* set qtime for current year and quarter */;
qtime_master=qtime;
qtime=.;
if multi_first_year~=. and multi_first_quarter~=. then do;
    %qtime(multi_first_year,multi_first_quarter) /* set qtime for the
first appearance of multi */;
end;
qtime_first=qtime;
qtime=.;
```

Data available on subunits for at least one variable in this SEIN YEAR QUARTER

```
data_avail=in_202=1 and ((impute_wage=0 and no_wages=0) or (impute_emp1=0
and no_emp1=0) or (impute_emp2=0 and no_emp2=0) or (impute_emp3=0 and
no_emp3=0));

special_handle=0;
** Condition 1 **;
if in_UI=1 and in_202=0 and ever_multi=1 then special_handle=1;
** Condition 2 **;
if in_UI=0 and in_202=1 and impute_data=1 and num_estabs>1 then
special_handle=2;
** Condition 3 and Condition 4 **;
if in_UI=1 and in_202=1 then do;
    if no_data=1 and multi_unit=1 then special_handle=3;
    if impute_data=1 and num_estabs>1 then special_handle=4;
end;
```

The get_XX variables reveal if there is information on wages, emp1, emp2, emp3 available in off qtrs

```

        get_wages=0;
        if special_handle=1 and ever_wages=1 and qtime_master>qtime_first then
get_wages=1;
        if special_handle=2 and impute_wage=1 and ever_wages=1 and
qtime_master>qtime_first then get_wages=1;
        if special_handle=3 and no_wages=1 and ever_wages=1 and
qtime_master>qtime_first then get_wages=1;
        if special_handle=4 and impute_wage=1 and ever_wages=1 and
qtime_master>qtime_first then get_wages=1;
        get_emp1=0;
        if special_handle=1 and ever_emp1=1 and qtime_master>qtime_first then
get_emp1=1;
        if special_handle=2 and impute_emp1=1 and ever_emp1=1 and
qtime_master>qtime_first then get_emp1=1;
        if special_handle=3 and no_emp1=1 and ever_emp1=1 and
qtime_master>qtime_first then get_emp1=1;
        if special_handle=4 and impute_emp1=1 and ever_emp1=1 and
qtime_master>qtime_first then get_emp1=1;
        get_emp2=0;
        if special_handle=1 and ever_emp2=1 and qtime_master>qtime_first then
get_emp2=1;
        if special_handle=2 and impute_emp2=1 and ever_emp2=1 and
qtime_master>qtime_first then get_emp2=1;
        if special_handle=3 and no_emp2=1 and ever_emp2=1 and
qtime_master>qtime_first then get_emp2=1;
        if special_handle=4 and impute_emp2=1 and ever_emp2=1 and
qtime_master>qtime_first then get_emp2=1;
        get_emp3=0;
        if special_handle=1 and ever_emp3=1 and qtime_master>qtime_first then
get_emp3=1;
        if special_handle=2 and impute_emp3=1 and ever_emp3=1 and
qtime_master>qtime_first then get_emp3=1;
        if special_handle=3 and no_emp3=1 and ever_emp3=1 and
qtime_master>qtime_first then get_emp3=1;
        if special_handle=4 and impute_emp3=1 and ever_emp3=1 and
qtime_master>qtime_first then get_emp3=1;

        no_get_data=get_wages=0 and get_emp1=0 and get_emp2=0 and get_emp3=0;
        if no_get_data=0 then output INTERWRK.special_handle_list_10;
end;

output INTERWRK.alldata_10;
run;

```

Get some statistics on which firms have data in the 202 that can be used . to determine firm structure .

```
/* diagnostics moved to ecfdiag_10.sas */
```

Get the Firm Histories for firms that are selected for special handling .

```

proc sort data=INTERWRK.special_handle_list_10 out=step1(keep=sein) nodupkey;
  by sein;
run;

data INTERWRK.special_handle_history_10(index=(sein_year_quarter=(sein year
quarter)));
  merge INTERWRK.alldata_10(in=a) step1(in=b);
  by sein;

  if a=1 and b=1 then output;
run;

/* diagnostics moved to ecfdiag_10.sas */

```

1.5.2.11 library/sasprogs/11_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/11_ecf.sas >--+ */
/*----- LEGACY NAME: 11_special_handle.sas -----*/
```

This program looks for subunit structure in off years and quarters.

For each SEIN YEAR QUARTER record selected for structure imputation I look in both previous and future quarters for a period where the firm reported payroll and/or employment for the SEINUNITs. The closest quarter is kept for further processing. In Illinois about 80% are found within +/- four quarters.

```
/* See runtime_ecf.sas for configuration options */
/* ALL macros are stored in runtime_ecf.sas */

/*****
*****/
/* REQUIRES-SAS: runtime_ecf.sas
*/
/* REQUIRES-DATA: special_handle_list_10.sas7bdat,
special_handle_history_10.sas7bdat */
/* PROVIDES-DATA: special_handle_11.sas7bdat
*/
/*****
*****/

* %include "runtime_ecf.sas";

title1 "Read in the ES202 from /data/master/ui/&state./YYYY";
title2 "Program Location: &curdir./11_special_handle.sas";

data INTERWRK.special_handle_11();
    set INTERWRK.special_handle_list_10;
```

Find the closest record with data. Only look up records that have a data structure in off years .

```
length seinunit $5 sein_ori $12;

qtime=qtime_master;
stop=0;
do i=1 to &qend while (stop=0) ;
    if (qtime+i)<(&qend+1) then do;
        qtime=qtime+i;
        %yqreal(qtime);
        continue=0;
        continue1=0;
        sein_ori=sein;
```

Initialize before lookup.

```

do while ( continue=0 );
year_found=.;
quarter_found=.;
total_wages=.;
empl_month1=.;
empl_month2=.;
empl_month3=.;
multi_unit=.;
data_avail=.;
seinunit=" ";
no_wages=.;
impute_wage=.;
no_emp1=.;
impute_emp1=.;
no_emp2=.;
impute_emp2=.;
no_emp3=.;
impute_emp3=.;
sein_wages=.;
sein_emp1=.;
sein_emp2=.;
sein_emp3=.;

```

Lookup the necessary info in the history file.

```

set INTERWRK.special_handle_history_10
(keep=sein year quarter seinunit multi_unit total_wages
empl_month1 empl_month2 empl_month3 data_avail
no_wages impute_wage no_emp1 impute_emp1
no_emp2 impute_emp2 no_emp3 impute_emp3
sein_wages sein_emp1 sein_emp2 sein_emp3
)
key=sein_year_quarter;
if continue1=1 then continue=1;
if _iorc_ = 0 then do;
if data_avail=1 and multi_unit=1 then do;
stop=1;
year_found=year;
quarter_found=quarter;
%yqreal(qtime_master);
output;
%yqreal(qtloop);
end;
end;
else do;
_error_=0;
sein="zzzzzzzzzz";
continue1=1;
end;
*put qtime_master qtloop sein year quarter year_found=
quarter_found= total_wages= empl_month1= empl_month2= empl_month3=;
end;
sein=sein_ori;
end;

```

Backward lookup? Initialize before lookup.

```

if (qtime-i)>0 and stop=0 then do;
  qtime=qtime-i;
  %yqreal(qtime);
  continue=0;
  continue1=0;
  sein_ori=sein;
do while ( continue=0 );
  year_found=.;
  quarter_found=.;
  total_wages=.;
  empl_month1=.;
  empl_month2=.;
  empl_month3=.;
  multi_unit=.;
  data_avail=.;
  seinunit=" ";
  no_wages=.;
  impute_wage=.;
  no_emp1=.;
  impute_emp1=.;
  no_emp2=.;
  impute_emp2=.;
  no_emp3=.;
  impute_emp3=.;
  sein_wages=.;
  sein_emp1=.;
  sein_emp2=.;
  sein_emp3=.;

```

Do lookup.

```

      set INTERWRK.special_handle_history_10
(keep=sein year quarter seinunit multi_unit total_wages
  empl_month1 empl_month2 empl_month3 data_avail
  no_wages impute_wage no_emp1 impute_emp1
  no_emp2 impute_emp2 no_emp3 impute_emp3
  sein_wages sein_emp1 sein_emp2 sein_emp3
)
key=sein_year_quarter;
  if continue1=1 then continue=1;
  if _iorc_ = 0 then do;
    if data_avail=1 and multi_unit=1 then do;
      stop=1;
      year_found=year;
      quarter_found=quarter;
      %yqreal(qtime_master);
      output;
      %yqreal(qtime);
    end;
  end;
  else do;
    _error_=0;
    sein="zzzzzzzzzz";
    continue1=1;
  end;
  *put qtime_master qtime sein year quarter year_found=
quarter_found= total_wages= empl_month1= empl_month2= empl_month3=;
  end;
  sein=sein_ori;
end;
if stop=0 then do;
  %yqreal(qtime_master);
  output;
end;
run;

```

Diagnostics moved to ecfdiag_11.sas

1.5.2.12 library/sasprogs/12_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/12_ecf.sas >--+ */
/*----- LEGACY NAME: 12_distribute.sas -----*/
```

This program uses the record structure identified in the previous program to allocate payroll and employment. Interleave the new record structure with the existing data.

The process of imputing the record structure of an SEIN in a given year and quarter is completed here. I use SEIN level information on employment and payroll from the CURRENT quarter combined with distribution of employment / payroll across the SEINUNIT's information learned from off quarter reports to allocate payroll and employment to the subunits. I combine the newly imputed record structure with the old data to create the final record structure.

NOTE: The best_flag variable when combined with the structure_fix variable can be used to identify the type of edits and data source of the best_xx variables.

The record structure file &st._leg_structure.sas7bdat is created here. This file is used to determine the final structure of the LEG and by the pred/succ programs.

```
/******
***/
/* REQUIRES-DATA: special_handle_11.sas7bdat
*/
/* PROVIDES-DATA: ss_employer_char_unit.sas7bdat, ecf_ss_leg_structure.sas7bdat
*/
/******
***/

* %include "runtime_ecf.sas";

title1 "Create the best_XXX variables";
title2 "Program Location: &curdir./12_distribute.sas";

data step0(keep=sein year quarter seinunit year_found quarter_found best_wages
best_emp1 best_emp2 best_emp3
emp1_202 emp2_202 emp3_202 wages_202 emp1_UI emp2_UI emp3_UI
wages_UI empl_month1 empl_month2 empl_month3 total_wages
sein_wages sein_emp1 sein_emp2 sein_emp3 special_handle in_202
in_UI source);
set INTERWRK.special_handle_11();
```

Allocate the SEIN level data to the Estabs:

WAGES

```
    if impute_wage=0 and no_wages=0 then do;
      if special_handle=1 or special_handle=3 then
best_wages=wages_UI*(total_wages/sein_wages);
      if special_handle=2 or special_handle=4 then
best_wages=wages_202*(total_wages/sein_wages);
      end;
    else if impute_emp2=0 and no_emp2=0 then do;
      if special_handle=1 or special_handle=3 then
best_wages=wages_UI*(empl_month2/sein_emp2);
      if special_handle=2 or special_handle=4 then
best_wages=wages_202*(empl_month2/sein_emp2);
      end;
    else if impute_emp1=0 and no_emp1=0 then do;
      if special_handle=1 or special_handle=3 then
best_wages=wages_UI*(empl_month1/sein_emp1);
      if special_handle=2 or special_handle=4 then
best_wages=wages_202*(empl_month1/sein_emp1);
      end;
    else if impute_emp3=0 and no_emp3=0 then do;
      if special_handle=1 or special_handle=3 then
best_wages=wages_UI*(empl_month3/sein_emp3);
      if special_handle=2 or special_handle=4 then
best_wages=wages_202*(empl_month3/sein_emp3);
      end;
```

EMP1

```
    if impute_emp1=0 and no_emp1=0 then do;
      if special_handle=1 or special_handle=3 then
best_emp1=emp1_UI*(empl_month1/sein_emp1);
      if special_handle=2 or special_handle=4 then
best_emp1=emp1_202*(empl_month1/sein_emp1);
      end;
    else if impute_emp2=0 and no_emp2=0 then do;
      if special_handle=1 or special_handle=3 then
best_emp1=emp1_UI*(empl_month2/sein_emp2);
      if special_handle=2 or special_handle=4 then
best_emp1=emp1_202*(empl_month2/sein_emp2);
      end;
    else if impute_emp3=0 and no_emp3=0 then do;
      if special_handle=1 or special_handle=3 then
best_emp1=emp1_UI*(empl_month3/sein_emp3);
      if special_handle=2 or special_handle=4 then
best_emp1=emp1_202*(empl_month3/sein_emp3);
      end;
    else if impute_wage=0 and no_wages=0 then do;
      if special_handle=1 or special_handle=3 then
best_emp1=emp1_UI*(total_wages/sein_wages);
      if special_handle=2 or special_handle=4 then
best_emp1=emp1_202*(total_wages/sein_wages);
      end;
```

EMP2

```

    if impute_emp2=0 and no_emp2=0 then do;
      if special_handle=1 or special_handle=3 then
best_emp2=emp2_UI*(empl_month2/sein_emp2);
      if special_handle=2 or special_handle=4 then
best_emp2=emp2_202*(empl_month2/sein_emp2);
      end;
    else if impute_emp1=0 and no_emp1=0 then do;
      if special_handle=1 or special_handle=3 then
best_emp2=emp2_UI*(empl_month1/sein_emp1);
      if special_handle=2 or special_handle=4 then
best_emp2=emp2_202*(empl_month1/sein_emp1);
      end;
    else if impute_emp3=0 and no_emp3=0 then do;
      if special_handle=1 or special_handle=3 then
best_emp2=emp2_UI*(empl_month3/sein_emp3);
      if special_handle=2 or special_handle=4 then
best_emp2=emp2_202*(empl_month3/sein_emp3);
      end;
    else if impute_wage=0 and no_wages=0 then do;
      if special_handle=1 or special_handle=3 then
best_emp2=emp2_UI*(total_wages/sein_wages);
      if special_handle=2 or special_handle=4 then
best_emp2=emp2_202*(total_wages/sein_wages);
      end;

```

EMP3

```

        if impute_emp3=0 and no_emp3=0 then do;
            if special_handle=1 or special_handle=3 then
best_emp3=emp3_UI*(empl_month3/sein_emp3);
            if special_handle=2 or special_handle=4 then
best_emp3=emp3_202*(empl_month3/sein_emp3);
            end;
        else if impute_emp2=0 and no_emp2=0 then do;
            if special_handle=1 or special_handle=3 then
best_emp3=emp3_UI*(empl_month2/sein_emp2);
            if special_handle=2 or special_handle=4 then
best_emp3=emp3_202*(empl_month2/sein_emp2);
            end;
        else if impute_emp1=0 and no_emp1=0 then do;
            if special_handle=1 or special_handle=3 then
best_emp3=emp3_UI*(empl_month1/sein_emp1);
            if special_handle=2 or special_handle=4 then
best_emp3=emp3_202*(empl_month1/sein_emp1);
            end;
        else if impute_wage=0 and no_wages=0 then do;
            if special_handle=1 or special_handle=3 then
best_emp3=emp3_UI*(total_wages/sein_wages);
            if special_handle=2 or special_handle=4 then
best_emp3=emp3_202*(total_wages/sein_wages);
            end;
run;

/* Modified and moved to ecf_diag12.sas
*proc print data=step0();
* id sein year quarter seinunit;
* var wages_202 emp1_202 emp2_202 emp3_202 emp1_UI emp2_UI emp3_UI wages_UI
total_wages empl_month1 empl_month2 empl_month3 best_wages best_emp1 best_emp2
best_emp3;
*run;
*/

data INTERWRK.step0_17(obs=1000);
    set step0(keep=wages_202 emp1_202 emp2_202 emp3_202 emp1_UI emp2_UI emp3_UI
wages_UI total_wages empl_month1 empl_month2 empl_month3
best_wages best_emp1 best_emp2 best_emp3
sein year quarter seinunit);
run;

```

The section below substantially modified : March 2003

A problem with seinsize_b was noticed during other work. seinsize_b was getting overwritten with the value from the donor record. Other variables were also affected leading to a small loss of information at the seinunit level

Record structure imputation can also change data for the current record even if a structure is present This occurs when special_handle in(2 3 4) usually the structure is similar in the closest off qtr but in rare cases there can be significant diffs

Prepare the alldata_10 dataset for the adjusted data by selecting records that were NOT adjusted.

```

proc sort data=step0(keep=sein year quarter) out=step1 nodupkey;
  by sein year quarter;
run;

data step2;
  merge INTERWRK.allldata_10(in=a) step1(in=b);
  by sein year quarter;

  structure_fix=0;
  if a=1 and b=0 then output;
run;

proc print data=step2(obs=100);
  title3 "Observations before interleave";
run;

```

Rebuild the adjusted record picking pieces from the old and new. Select Variables to attach from the old record. Attach variables that differ by seinunit within sein

```

data step2_5;
merge INTERWRK.special_handle_history_10
      (in=a keep=sein year quarter seinunit
        empl_month1 empl_month2 empl_month3 total_wages
        county ein ein_bad ein_defect valid_ein
        owner_code sic sic_invalid auxiliary_code
        naics1997 naics_aux1997 naics2002 naics_aux2002
        naics_1997_invalid naics_2002_invalid
        naics_aux_1997_invalid naics_aux_2002_invalid
      rename=(
        county=county_old
        ein=ein_old
        ein_bad=ein_bad_old
        ein_defect=ein_defect_old
        valid_ein=valid_ein_old
        owner_code=owner_code_old
        sic=sic_old
        sic_invalid=sic_invalid_old
        auxiliary_code=auxiliary_code_old
        naics1997=naics1997_old
        naics2002=naics2002_old
        naics_aux1997=naics_aux1997_old
        naics_aux2002=naics_aux2002_old
        naics_1997_invalid=naics_1997_invalid_old
        naics_2002_invalid=naics_2002_invalid_old
        naics_aux_1997_invalid=naics_aux_1997_invalid_old
        naics_aux_2002_invalid=naics_aux_2002_invalid_old
      ))

      step0
      (in=b keep=sein year quarter year_found quarter_found seinunit
        best_wages best_emp1 best_emp2 best_emp3
        emp1_202 emp2_202 emp3_202 wages_202
        emp1_UI emp2_UI emp3_UI wages_UI
        in_202 in_UI source special_handle);

by sein year quarter seinunit;

if empl_month1<0 then empl_month1=0;
if empl_month2<0 then empl_month2=0;
if empl_month3<0 then empl_month3=0;
if total_wages<0 then total_wages=0;

if b=1 then output;
run;

proc print data=step2_5(obs=100);
run;

```

Attach the variables that are constant within SEIN First get the history dataset into SEIN YEAR QUARTER RECORDS.

```

data history_sein_06;
  set INTERWRK.special_handle_history_10
    (keep=sein year quarter yr_qtr
      data_avail ever_202 ever_UI
      ever_emp1 ever_emp2 ever_emp3
      ever_multi ever_wages
      impute_data impute_emp1 impute_emp2 impute_emp3
      impute_wage multi_first_quarter
      multi_first_year no_county no_data no_emp1 no_emp2 no_emp3
      no_get_data
      no_naics1997 no_naics2002 no_naics_aux1997 no_naics_aux2002
      no_sic no_wages payroll qtime_first qtime_master
      sein_emp1 sein_emp2 sein_emp3
      sein_wages seinsize_b seinsize_e seinsize_m
    );
  by sein year quarter;

  if first.quarter then output;
run;

data step3;
  merge history_sein_06(in=a) step2_5(in=b);

  by sein year quarter;

  if a=1 and b=1 then output;
run;

```

Select Variables to attach from the new record

```

proc sort data=step3;
  by sein year_found quarter_found seinunit;
run;

data step4;
  merge INTERWRK.special_handle_history_10
  (in=a rename=(year=year_found quarter=quarter_found)
  keep=sein year quarter yr_qtr seinunit NUM_ESTABS
  seinunit_bad seinunit_type multi_unit
  county ein ein_bad ein_defect valid_ein
  owner_code sic sic_invalid auxiliary_code
  naics1997 naics_aux1997 naics2002 naics_aux2002
  naics_1997_invalid naics_2002_invalid
  naics_aux_1997_invalid naics_aux_2002_invalid
  multi_unit_code master_multi_unit_code
  master_empl_month1_flg master_empl_month2_flg
  master_empl_month3_flg master_total_wages_flg
  empl_month1_flg empl_month2_flg
  empl_month3_flg total_wages_flg
  rename=(county=county_new
  ein=ein_new
  ein_bad=ein_bad_new
  ein_defect=ein_defect_new
  valid_ein=valid_ein_new
  owner_code=owner_code_new
  sic=sic_new
  sic_invalid=sic_invalid_new
  auxiliary_code=auxiliary_code_new
  naics1997=naics1997_new
  naics2002=naics2002_new
  naics_aux1997=naics_aux1997_new
  naics_aux2002=naics_aux2002_new
  naics_1997_invalid=naics_1997_invalid_new
  naics_2002_invalid=naics_2002_invalid_new
  naics_aux_1997_invalid=naics_aux_1997_invalid_new
  naics_aux_2002_invalid=naics_aux_2002_invalid_new
  ))

  step3 (in=b);
  by sein year_found quarter_found seinunit;

  length naics1997 naics2002 naics_aux1997 naics_aux2002 $6
  auxiliary_code owner_code
  naics_1997_invalid naics_2002_invalid
  naics_aux_1997_invalid naics_aux_2002_invalid
  sic_invalid $1
  sic $4
  ein $9
  county $3;

  structure_fix=1;
  best_flag=12;

```

Clean up AUXILIARY_CODE NAICS SIC EIN and COUNTY .

```

  auxiliary_code=auxiliary_code_old;
  if auxiliary_code_old=" " then auxiliary_code=auxiliary_code_new;

  county=county_old;
  if county_old="ZZZ" or county_old=" " then county=county_new;

```

EIN

```
ein=ein_old;
ein_bad=ein_bad_old;
ein_defect=ein_defect_old;
valid_ein=valid_ein_old;
if ein_defect_old>0 or ein_defect=. then do;
    ein=ein_new;
    ein_bad=ein_bad_new;
    ein_defect=ein_defect_new;
    valid_ein=valid_ein_new;
end;
```

Owner_code

```
owner_code=owner_code_old;
if owner_code_old="9" or owner_code_old=" " then owner_code=owner_code_new;
```

SIC

```
sic=sic_old;
sic_invalid=sic_invalid_old;
if sic_old="9999" or sic_old=" " then do;
    sic=sic_new;
    sic_invalid=sic_invalid_new;
end;
```

NAICS1997

```
naics1997=naics1997_old;
naics_1997_invalid=naics_1997_invalid_old;
if naics1997_old="999999" or naics1997_old=" " then do;
    naics1997=naics1997_new;
    naics_1997_invalid=naics_1997_invalid_new;
end;
```

NAICS2002

```
naics2002=naics2002_old;
naics_2002_invalid=naics_2002_invalid_old;
if naics2002_old="999999" or naics2002_old=" " then do;
    naics2002=naics2002_new;
    naics_2002_invalid=naics_2002_invalid_new;
end;
```

NAICS_AUX

```

naics_aux1997=naics_aux1997_old;
naics_aux_1997_invalid=naics_aux_1997_invalid_old;
if naics_aux1997_old="999999" or naics_aux1997_old=" " then do;
    naics_aux1997=naics_aux1997_new;
    naics_aux_1997_invalid=naics_aux_1997_invalid_new;
end;

naics_aux2002=naics_aux2002_old;
naics_aux_2002_invalid=naics_aux_2002_invalid_old;
if naics_aux2002_old="999999" or naics_aux2002_old=" " then do;
    naics_aux2002=naics_aux2002_new;
    naics_2002_invalid=naics_2002_invalid_new;
end;

if a=1 and b=1 then output;
run;

proc sort data=step4;
    by sein year quarter seinunit;
run;

proc contents;
run;

proc print data=step4(obs=100);
title3 "First 100 Records with updated structure";
run;

```

Interleave step4 and step2 back together .

```

data step5
  (keep=sein year quarter yr_qtr seinunit NUM_ESTABS
   SIC sic_invalid owner_code emp1_UI emp2_UI emp3_UI wages_UI
   auxiliary_code naics1997 naics2002 naics_aux1997 naics_aux2002
   empl_month1_flg empl_month2_flg empl_month3_flg
   total_wages_flg master_empl_month1_flg master_empl_month2_flg
   master_empl_month3_flg master_total_wages_flg
   master_multi_unit_code multi_unit_code naics_1997_invalid
   naics_2002_invalid naics_aux_1997_invalid naics_aux_2002_invalid
   best_wages best_emp1 best_emp2 best_emp3 best_flag county
   ein ein_bad ein_defect valid_ein
   empl_month1 empl_month2 empl_month3 total_wages
   ever_202 ever_UI ever_emp1 ever_emp2 ever_emp3 ever_multi
   ever_wages
   in_202 in_UI
   multi_first_quarter multi_first_year multi_unit
   seinunit_bad seinunit_type source special_handle
   sein_wages sein_emp1 sein_emp2 sein_emp3 payroll
   seinsize_m seinsize_e seinsize_b structure_fix
  );
set step4 step2;
  by sein year quarter seinunit;

if structure_fix=1 then do;
  sein_emp1=emp1_202;
  sein_emp2=emp2_202;
  sein_emp3=emp3_202;
  sein_wages=wages_202;
  if sein_emp1=. then sein_emp1=0;
  if sein_emp2=. then sein_emp2=0;
  if sein_emp3=. then sein_emp3=0;
  if sein_wages=. then sein_wages=0;
end;
run;

proc contents;
run;

proc freq;
  tables best_flag*structure_fix special_handle;
run;

proc print data=step5(obs=100);
title3 "Final File";
run;

```

THESE ARE QA INSTRUCTIONS: Check the data to make sure all sein year quarters are in the dataset. The number of obs should equal the number of obs in the dataset step2 in the program 04_sein_totals.sas Sum the best_XX variables for each SEIN YEAR QUARTER .

```

data step6(keep=sein year quarter sein_best_wages sein_best_emp1 sein_best_emp2
sein_best_emp3);
  set step5(keep=sein year quarter best_wages best_emp1 best_emp2 best_emp3);
  by sein year quarter;

  if first.quarter then do;
    sein_best_wages=0;
    sein_best_emp1=0;
    sein_best_emp2=0;
    sein_best_emp3=0;
  end;

  sein_best_wages+best_wages;
  sein_best_emp1+best_emp1;
  sein_best_emp2+best_emp2;
  sein_best_emp3+best_emp3;

  if last.quarter then output;
run;

proc print data=step6(obs=100);
title3 "SEIN YEAR QUARTER summaries";
run;

```

Attach the SEIN YEAR QUARTER summary info to the SEINUNIT file .

```

data INTERWRK.&state._employer_char_unit
    (sortedby=sein year quarter seinunit)
/*
*   INTERWRK.ecf_&state._leg_structure
*   (keep=sein year quarter seinunit sic best_emp1 county
*   sortedby=sein year quarter seinunit)
*/
;
merge step5 step6;
    by sein year quarter;

    length
    best_emp1 4
    best_emp2 4
    best_emp3 4
    best_flag 3
    best_wages 5
    ein_bad 3
    ein_defect 3
    emp1_UI 4
    emp2_UI 4
    emp3_UI 4
    wages_UI 5
    ever_202 3
    ever_UI 3
    ever_emp1 3
    ever_emp2 3
    ever_emp3 3
    ever_multi 3
    ever_wages 3
    in_202 3
    in_UI 3
    multi_first_quarter 3
    multi_first_year 3
    multi_unit 3
    num_estabs 4
    sein_best_emp1 4
    sein_best_emp2 4
    sein_best_emp3 4
    sein_best_wages 5
    sein_emp1 4
    sein_emp2 4
    sein_emp3 4
    sein_wages 5
    seinsize_m 4
    seinsize_e 4
    seinsize_b 4
    payroll 5
    seinunit_bad 3
    seinunit_type 3
    total_wages 5
    source 3
    special_handle 3
    structure_fix 3
    valid_ein 3
;
run;

/* diagnostics moved to ecf_diag12.sas */

```

1.5.2.13 library/sasprogs/13_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/13_ecf.sas >--+ */
/* OLDNAME: leg/1.1.04/library/sasprogs/01_leg.sas */
```

The LEG contains complete detailed geography for the universe of sein seinunit year quarter observations defined by the ECF. The geography information used is from the GAL with a series of longitudinal edits applied.

```
/*=====*/
/* 01.01.base.sas */
/* */
/* DESCRIPTION */
/* */
/* Merge es202, gal xwalk, and ecf. Stack files. */
/* Merge with gal. */
/* */
/* REQUIRES-SAS: config.sas */
/* */
/* REQUIRES-DATA: esYYYY.sas7bdat */
/* REQUIRES-DATA: SS_leg_structure.sas7bdat */
/* REQUIRES-DATA: xwlk_eYYYY.sas7bdat */
/* REQUIRES-DATA: gal_SS.sas7bdat */
/* */
/* PROVIDES-DATA: esgal.sas7bdat */
/*=====*/

** %include 'config.sas';

options mprint;
```

The first part merges all available ES202, GAL and ex-LEG_STRUCTURE. If a GAL file does not exist, then a pseudo-file is created.

```
%create_esgal;
```

Merge above file with the concatenated GAL (here gal_ss) and county centroid file. Keep geography info.

```
proc sort data=esgal;
  by galid;
run;

/* introduced in 3.1.12 */
%choose_gal;

data INTERWRK.esgal
  (keep=es_galid es_geo es_geoqual quarter sein seinunit year es_sic);
  merge
    esgal
    (in=in1)
    gal
    (in=in2)
  ;
  length es_geoqual 3.;
  by galid;
  if in1;
  if in1 then flag_mtch=1;
  if in1 and in2 then flag_mtch=3;
```

Processing for missing GALID. GALID is empty when the entity locked address info (no GALID was generated on the GAL). GEOQUAL=9 means the entity failed to geocode. These codes are expanded here.

```
if galid='' or a_geoqual=9 then do;
```

If we have county information on the ES202, we point to county centroid.

```
  if es_county~='' then do;
    es_galid="A"||put(stfips("&state"),z2.)||es_county||"999999999";
    es_geoqual=9;
    es_geo=put(stfips("&state"),z2.)||es_county||"0000000";
    end; /* end centroid loop */
  else do;
```

If no county information is available, we prepare for the impute. If industry info is available, then we will impute based on SIC. Otherwise, if neither county nor industry info is available, we impute unconditionally.

```
  if es_sic~='' then do;
    es_galid='';
    es_geoqual=10;
    es_geo='';
    end; /* end SIC loop */
  else do;
    es_galid='';
    es_geoqual=11;
    es_geo='';
    end; /* end unconditional code */
  end; /* end non-centroid loop */
end; /* end geoqual 9/missing GALID loop */
```

Processing for non-missing GALID. First loop is for physical address.

```
  else do;
    if e_flag='P' then do;
      if substr(a_geo,1,2)=put(stfips("&state"),z2.) then do;
        es_galid=galid;
        es_geoqual=a_geoqual;
        es_geo=a_geo;
        end; /* end of loop with GALID flag P */
```

This should be non-operative (failsafe) code since at least GAL 3.x, since the GAL does not generate out-of-state GALID.

```
  else do; /* out of state loop */
    if es_county~='' then do;

    es_galid="A"||put(stfips("&state"),z2.)||es_county||"999999999";
    es_geoqual=9;
    es_geo=put(stfips("&state"),z2.)||es_county||"0000000";
    end; /* end centroid loop */
    else do;
      if es_sic~='' then do;
        es_galid='';
        es_geoqual=10;
        es_geo='';
        end; /* end SIC loop */
      else do;
        es_galid='';
        es_geoqual=11;
        es_geo='';
        end; /* end unconditional code */
      end; /* end of non-centroid loop */
    end; /* end of out of state loop */
  end; /* end P flag */
```

If no physical address used, then mailing address used. The geoquality indicator is increased (quality decreased), giving preference to physical location.

```
  else if e_flag='M' then do;
```

Mailing address is used only when it agrees with county on ES202.

```

        if es_county='' or es_county=substr(a_geo,3,3) then do;
            es_galid=galid;
            es_geoqual=a_geoqual+4;
            es_geo=a_geo;
            end; /* end county missing loop */

```

If county does not agree, then put in county centroid of ES202 county, overriding any GAL coding. This forces the geography of MAILING address-coded entities to correspond to their ES202 county.

```

        else do;
            es_galid="A"||put(stfips("&state"),z2.)||es_county||"999999999";
            es_geoqual=9;
            es_geo=put(stfips("&state"),z2.)||es_county||"0000000";
            end; /* county available loop */
        end; /* end of M flag loop */
    end; /* end of non-missing GALID loop */
run;

```

1.5.2.14 library/sasprogs/14_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/sasprogs/14_ecf.sas >--- */
/* OLDNAME: leg/1.1.04/library/sasprogs/02_leg.sas */
/* Time-stamp: <04/10/27 19:56:26 vilhuber> */

```

This program takes long file from step 1 and creates a wide file. Within each seinunit, geography in consecutive quarters is compared to determine whether or not a move took place. Then, within non-move time periods, the best geography is determined for each seinunit and applied over the time period. Seinunits for which no geography information is available are output to a separate impute dataset.

```

/*=====*/
/* 02.01.wide.sas */
/* */
/* DESCRIPTION */
/* */
/* Create wide file, do longitudinal geo improvements */
/* Output long file */
/* */
/* REQUIRES-SAS: config.sas */
/* */
/* REQUIRES-DATA: esgal.sas7bdat */
/* */
/* PROVIDES-DATA: leg.sas7bdat */
/* PROVIDES-DATA: impute.sas7bdat */
/*=====*/

```

```

proc sort data=INTERWRK.esgal
    out=esgal(keep=sein seinunit year quarter es_geo
              es_geoqual es_galid es_sic);
    by sein seinunit year quarter;
run;

```

```

data
  INTERWRK.impute
    (keep= sein seinunit es_sic1-es_sic&leg_qrange lgq1-lgq&leg_qrange)
  INTERWRK.leg_02
    (keep= sein seinunit year quarter es_galid es_geoqual es_geo leg_galid
      leg_geoqual flag_leggeo leg_geo move es_sic t_move);
  set esgal;
  by sein seinunit year quarter;

  length
    eg1-eg&leg_qrange $12.
    ea1-ea&leg_qrange $15.
    eq1-eq&leg_qrange 3.
    move1-move&leg_qrange 3.
    es_sic1-es_sic&leg_qrange $2.
    leg_geoqual 3.
    flag_leggeo 3.
    move 3.
    t_move 3.
    lgal1-lgal&leg_qrange $15.
    lg1-lg&leg_qrange $12.
  ;
  retain
    eg1-eg&leg_qrange
    cz1-cz&leg_qrange
    ea1-ea&leg_qrange
    eq1-eq&leg_qrange
    move1-move&leg_qrange
    flag_test1-flag_test&leg_qrange
    es_sic1-es_sic&leg_qrange
  ;

```

Create arrays. Note the use of leg_qrange, defined in the parameters file.
 The aes_ prefaced variables come from the GAL.

```

array aes_geo(&leg_qrange)    eg1      -eg&leg_qrange;
array aes_galid(&leg_qrange)  ea1      -ea&leg_qrange;
array aes_geoqual(&leg_qrange) eq1      -eq&leg_qrange;
array amove(&leg_qrange)      move1    -move&leg_qrange;
array aes_sic(&leg_qrange)    es_sic1  -es_sic&leg_qrange;
array aflag_test(&leg_qrange) flag_test1-flag_test&leg_qrange;

```

The leg_ prefaced variables are reassigned based on determination of a move, otherwise carried forward.

```

array aleg_galid{&leg_qrange} lgal1  -lgal&leg_qrange;
array aleg_geoqual{&leg_qrange} lgq1  -lgq&leg_qrange;
array aflag_leggeo{&leg_qrange} lgf1  -lgf&leg_qrange;
array aleg_geo{&leg_qrange}    lg1    -lg&leg_qrange;

array at_move{&leg_qrange}    t_move1-t_move&leg_qrange;

```

Create wide arrays of key variables

```

if first.seinunit then do;
  do i=1 to &leg_qrange;
    aes_geo(i)='';
    aes_galid(i)='';
    aes_geoqual(i)=.;
    amove(i)=.;
    aes_sic(i)='';
  end;
end;

iqtr=(year-&start_year)*4 + quarter;
aes_geo(iqtr)=es_geo;
aes_galid(iqtr)=es_galid;
aes_geoqual(iqtr)=es_geoqual;
aes_sic(iqtr)=es_sic;

```

Determine if a new address is a move or a change in address quality. A empty geoquality condition never triggers a move.

```

if aes_geoqual(iqtr)='' then do;
  amove(iqtr)=.;
  aflag_test(iqtr)=17;
end;

```

The first quarter always defines the beginning of a move period

```

else if iqtr=1 then do;
  amove(iqtr)=1;
  aflag_test(iqtr)=16;
end;
else do; /* all other quarters */

```

If geography info is not available in $t - 1$ but is available in t , then define t as the beginning of the move period.

MARC: Forward-fill hole only? No backfilling of holes?

```

if aes_geoqual(iqtr-1)=. and aes_geoqual(iqtr)~=. then do;
  amove(iqtr)=1;
  aflag_test(iqtr)=1;
end;

```

Imputed geography is never used to determine a move.

MARC: This statement is not entirely true, since geoqual=9 is imputed geography!

```

else if /*aes_geoqual(iqtr-1)>=10 or*/ aes_geoqual(iqtr)>=10 then do;
  amove(iqtr)=0;
  aflag_test(iqtr)=2;
end;
else if aes_geoqual(iqtr-1)>=10 then do;
  amove(iqtr)=1;
  aflag_test(iqtr)=99;
end;
else do;

```

If the GALID is the same then there is no move.

```

if aes_galid(iqtr-1)=aes_galid(iqtr) then do;
  amove(iqtr)=0;
  aflag_test(iqtr)=3;
end;
else do;

```

Now we know that the GALID is different. Use physical address information first. A change in geography that is due to a change from physical to mailing address will not be considered a move. This is only done if SOME adequate geo information is available over the period.

Note that the previous module re-coded all mailing address geoqual with a +4, thus excluding geocoding based on mailing address from the condition below.

```

if min(of eq1-eq&leg_qrange)<=4 then do;

```

In order for there to be a geography move for an seinunit, the GALIDS must be different in a way that is not due to variations in the quality of geography. For example, a rooftop and a block group geocode will always necessarily have different GALIDS. However, there will only be a move marked if the rooftop and the block group are in different block groups. If they are in the same block group, the difference in GALIDS is only considered a change in geography quality, not a move.

```

/* Loop if at least blockface geoquality */
if max(aes_geoqual(iqtr-1), aes_geoqual(iqtr))<=2 then do;

```

This condition handles the case when the block group differs.

```

if aes_geo(iqtr-1)~=aes_geo(iqtr) then do;
  amove(iqtr)=1;
  aflag_test(iqtr)=4;
end;

```

If the block group does NOT differ and the geoquality indicators are the same, then a move is present.

MARC: Possibly doubtful..

```

else do;
  if aes_geoqual(iqtr-1)=aes_geoqual(iqtr) then do;
    aflag_test(iqtr)=5;
    amove(iqtr)=1;
    end; /* end equal geoqual, same block group */

```

Finally, if the geoquality is different, and the block is the same, then it is NOT considered a move.

```

else do;
  amove(iqtr)=0;
  aflag_test(iqtr)=6;
  end; /* end same block, different geoqual */
end;
end;

```

Next condition: geo quality is at least block group.

```

if max(aes_geoqual(iqtr-1), aes_geoqual(iqtr))=3 then do;

```

First, if the block group differs, then a move is assumed.

```

if
  substr(aes_geo(iqtr-1),1,12)~=substr(aes_geo(iqtr),1,12) then do;
  amove(iqtr)=1;
  aflag_test(iqtr)=7;
end;

```

If the block group is the same (block group the same, GALID different), then we compare the geo quality. If the geo quality is the same (but at least block group quality), then a move is assumed.

```

else do; /* same block group */
  if aes_geoqual(iqtr-1)=aes_geoqual(iqtr) then do;
    amove(iqtr)=1;
    aflag_test(iqtr)=8;
  end; /* end same geoqual */

```

Finally, if the block group is the same, but the geo quality differs, then no move is assumed to occur, the difference in GALIDs stemming from changes in quality, not location.

```

else do; /* different geo quality */
  amove(iqtr)=0;
  aflag_test(iqtr)=9;
end; /* end different geo quality */
end; /* end same block group */
end; /* end geo quality = block group */

```

Next condition: geo quality is at least tract level.

```

if max(aes_geoqual(iqtr-1), aes_geoqual(iqtr))=4 then do;

```

If the tract differs, then a move is assumed.

```

  if
  substr(aes_geo(iqtr-1),1,11)~=substr(aes_geo(iqtr),1,11) then do;
    amove(iqtr)=1;
    aflag_test(iqtr)=10;
  end;

```

Else (geo quality of one of the observations at least tract level, and the tract level is the same), then no move occurs.

MARC: this implies that geo quality is the SAME.

```

else do;
  amove(iqtr)=0;
  aflag_test(iqtr)=12;
end;
end; /* end of geoqual = 4 condition */
end; /* end of physical address condition */
else do; /* start of mailing address (geoqual > 4) condition */

```

Repeat the above procedure for mailing addresses. Cases of aflag_test() in (4-12) occur the same as for physical addresses. Exception is a comparison of county in imputed cases (aflag_test=13).

Mailing address is at least block face quality

```

if max(aes_geoqual(iqtr-1), aes_geoqual(iqtr))<=6 then do;

```

This condition handles the case when the block group differs.

```

  if aes_geo(iqtr-1)~=aes_geo(iqtr) then do;
    amove(iqtr)=1;
    aflag_test(iqtr)=4;
  end;

```

If the block group does NOT differ and the geoquality indicators are the same, then a move is present.

```

else do;
  if aes_geoqual(iqtr-1)=aes_geoqual(iqtr) then do;
    aflag_test(iqtr)=5;
    amove(iqtr)=1;
  end; /* end equal geoqual, same block group */

```

Finally, if the geoquality is different, and the block is the same, then it is NOT considered a move.

```

else do;
  amove(iqtr)=0;
  aflag_test(iqtr)=6;
end; /* end same block, different geoqual */
end;
end;

```

Next condition: geo quality is at least block group.

```

if max(aes_geoqual(iqtr-1), aes_geoqual(iqtr))=7 then do;

```

First, if the block group differs, then a move is assumed.

```

  if
  substr(aes_geo(iqtr-1),1,12)~=substr(aes_geo(iqtr),1,12) then do;
    amove(iqtr)=1;
    aflag_test(iqtr)=7;
  end;

```

If the block group is the same (block group the same, GALID different), then we compare the geo quality. If the geo quality is the same (but at least block group quality), then a move is assumed.

```

else do; /* same block group */
  if aes_geoqual(iqtr-1)=aes_geoqual(iqtr) then do;
    amove(iqtr)=1;
    aflag_test(iqtr)=8;
  end; /* end same geoqual */

```

Finally, if the block group is the same, but the geo quality differs, then no move is assumed to occur, the difference in GALIDs stemming from changes in quality, not location.

```

else do; /* different geo quality */
  amove(iqtr)=0;
  aflag_test(iqtr)=9;
end; /* end different geo quality */
end; /* end same block group */
end; /* end geo quality = block group */

```

Next condition: geo quality is at least tract level.

```

if max(aes_geoqual(iqtr-1), aes_geoqual(iqtr))=8 then do;

```

If the tract differs, then a move is assumed.

```

  if
  substr(aes_geo(iqtr-1),1,11)~=substr(aes_geo(iqtr),1,11) then do;
    amove(iqtr)=1;
    aflag_test(iqtr)=10;
  end;

```

Else (geo quality of one of the observations at least tract level, and the tract level is the same), then no move occurs.

MARC: this implies that geo quality is the SAME.

```

else do;
  amove(iqtr)=0;
  aflag_test(iqtr)=12;
end;
end;
end;

```

This condition (13) is distinct for mailing address. If geo quality is IMPUTED (??) / at the county level then check the county.

```

if max(aes_geoqual(iqtr-1), aes_geoqual(iqtr))=9 then do;

```

If the county is different, then assume move.

```

then do;
    if substr(aes_geo(iqtr-1),1,5)~=substr(aes_geo(iqtr),1,5)
        amove(iqtr)=1;
        aflag_test(iqtr)=13;
        end;
    else do;

```

This case should never occur b/c if both geoquals=5 and they refer to the same county, they would have had the same galids.

```

        if aes_geoqual(iqtr-1)=aes_geoqual(iqtr) then do;
            amove(iqtr)=1;
            aflag_test(iqtr)=14;
            end;
        else do;
            amove(iqtr)=0;
            aflag_test(iqtr)=15;
            end;
        end; /* end of county is the same condition */
    end; /* end of geoqual=9 condition */
end; /* end of mailing address condition */
end; /* end of GALID unequal condition */
end; /* end of positive geoqual, non-first period */

```

Summing up the values of aflag_test(), the condition of comparison between geo codes, geo quality in t and $t - 1$, and their relationship with the move flag:

```

/*=====
aflag_test =                                amove =
 1 : Geography not available in t-1          1
 2 : Geography in t imputed (geoqual>=10)   0
 3 : GALID the same                          0
-----
 4 : Quality at least blockface
     different block group                    1
 5 : Quality at least blockface
     same block group, same quality          1
 6 : Quality at least blockface
     same block group, diff quality         1
-----
 7 : Quality at least block group
     different block group                    1
 8 : Quality at least block group
     same block group, same quality          1
 9 : Quality at least block group
     same block group, diff quality         1
-----
10 : Quality at least tract level
     different tract level                    1
11 : not defined
-----
12 : Quality at least tract level
     same tract level                        1
-----
13 : Specific to mailing address:
     Geoquality is at least county level,
     county different                        1
-----
14 : Geoqual at least county,
     first 5 char of geo are same
     geoquality the same.
     (note: should not occur, because condition
     implies that the same GALID is assigned.) 1
15 : Geoqual at least county,
     first 5 char of geo are same
     geoquality the differs                  0
-----
16 : First quarter                          1
-----
17 : Empty geoquality                        .
-----
99 : Geography in t-1 imputed (geoqual>=10) 1
=====*/

```

Using move information, define best available geography, i.e. leg variables.

```

if last.seinunit then do;
  do i=1 to &leg_qrange;
    aleg_galid(i)=" ";
    aleg_geoqual(i)=.;
    aflag_leggeo(i)=.;
    aleg_geo(i)=" ";
  end;
do i=1 to &leg_qrange;
  im1=i-1;
  if aes_geoqual(i)=. then do;
    aleg_galid(i)='';
    aleg_geoqual(i)=.;
    aflag_leggeo(i)=.;
    aleg_geo(i)='';
    at_move(i)=0;
  end;
end;

```

If there is a move then assign new geography (GALID, GEOQUAL, GEOCODE).

```

else if amove(i)=1 then do;
  at_move(i)=0;
  aleg_galid(i)=aes_galid(i);
  aleg_geoqual(i)=aes_geoqual(i);
  aflag_leggeo(i)=0;
  aleg_geo(i)=aes_geo(i);
end;
else do;

```

If there is not a move then compare current and last period geography quality. Apply the better quality geography to all quarters within the seinunit, non-move time period.

```

    at_move(i)=at_move(im1)+1;
    if aleg_geoqual(im1)>aes_geoqual(i) then do;
        new_best=aes_galid(i);
        new_bestq=aes_geoqual(i);
        new_bestg=aes_geo(i);
        back=i-at_move(i);
        do j=back to i;
            aleg_galid(j)=new_best;
            aleg_geoqual(j)=new_bestq;
            aflag_leggeo(j)=at_move(i)-j+back;
            aleg_geo(j)=new_bestg;
        end;
    end;
    else if aleg_geoqual(im1)=aes_geoqual(i) then do;
        aleg_galid(i)=ales_galid(im1);
        aleg_geoqual(i)=ales_geoqual(im1);
        aflag_leggeo(i)=0;
        aleg_geo(i)=ales_geo(im1);
    end;
    else if aleg_geoqual(im1)<aes_geoqual(i) then do;
        aleg_galid(i)=ales_galid(im1);
        aleg_geoqual(i)=ales_geoqual(im1);
        aleg_geo(i)=ales_geo(im1);
        j=i;
        do until (ales_galid(j)=aes_galid(j));    **j=i to 1;
            aflag_leggeo(i)=j-i-1;
            j=j-1;
        end;
    end;
end;
end;
do i=1 to &leg_qrange;
    if aleg_geoqual(i) in (10 11) then impute=1;
end;

```

Seinunits for which no geography information is available are output to a separate impute dataset.

```

if impute=1 then output INTERWRK.impute;

```

The others are now output to a "narrow" file.

```

do i=1 to &leg_qrange;
    leg_galid=ales_galid(i);
    leg_geoqual=ales_geoqual(i);
    leg_geo=ales_geo(i);
    move=amove(i);
    es_galid=aes_galid(i);
    es_geoqual=aes_geoqual(i);
    flag_leggeo=aflag_leggeo(i);
    es_geo=aes_geo(i);
    flag_test=aflag_test(i);
    t_move=at_move(i);
    year=floor((i-1)/4+&start_year);
    quarter=i-(year-&start_year)*4;
    es_sic=aes_sic(i);
    if aes_geoqual(i)~= . then output INTERWRK.leg_02;
end;
end;

run;

```

1.5.2.15 library/sasprogs/15_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/15_ecf.sas >--+ */
/* OLDNAME: leg/1.1.04/library/sasprogs/03_leg.sas */
/* Time-stamp: <04/08/26 12:05:06 vilhu001> */
```

This program takes the ECF and develops a set of probabilities based on SIC and employment for county location. These probabilities are then merged onto the IMPUTE dataset from step 2. The imputed county is then assigned based on employment shares by SIC, if available, or employment shares only. The IMPUTE file is then merged with the other data.

```
/*=====*/
/* 03.01.impute.sas */
/* */
/* DESCRIPTION */
/* */
/* If an seinunit never has address or county info, impute */
/* county. Impute based on distribution by employment by sic, */
/* if sic available, or overall distribution of employment. */
/* */
/* REQUIRES-SAS: config.sas */
/* */
/* REQUIRES-DATA: impute.sas7bdat */
/* REQUIRES-DATA: SS_leg_structure.sas7bdat */
/* REQUIRES-DATA: leg.sas7bdat */
/* */
/* PROVIDES-DATA: leg.sas7bdat */
/*=====*/
%let fail=; * include on 06-06-2003 to prevent the job from failing;

options mprint mlogic symbolgen;
```

Find distribution of employment by SIC and overall using an intermediate version of the ECF.

```

%macro impute(test=);

    data ecf(keep=sein seinunit best_emp1 es_sic county);
/* replaced in ECF 3.1.32 */
    /* set OUTPUTS.ecf_&state._leg_structure; */
    set INTERWRK.&state._employer_char_unit
    (keep=sein year quarter seinunit sic best_emp1 county)
    ;
    length es_sic $2.;
    es_sic=substr(sic,1,2);
run;

proc summary data=ecf (where=(county~='ZZZ'));
class es_sic county;
var best_emp1;
output out=counts sum=;
run;

proc sort data=counts out=temp;
by county;
run;

data temp;
set temp (keep=county);
by county;
if last.county then output;
run;

data _null_;
set temp nobs=totobs;
call symput('noc',totobs);
run;

%let noc=%eval(&noc-1);
%put Number of Counties=&noc.;

data num(keep=es_sic county county_emp)
den(keep=es_sic es_sic_emp);
set counts;
if _type_ in (0,2) then do;
es_sic_emp=best_emp1;
output den;
end;
if _type_ in (1,3) then do;
county_emp=best_emp1;
output num;
end;
run;

```

Create wide file of probabilities (CDF) and associated counties;

```

data probs (keep=es_sic county p1-p&noc cty1-cty&noc es_sic_emp
ctyemp1-ctyemp&noc.);
merge
  num(in=in1)
  den;
by es_sic;
if in1;
length
  p1-p&noc 8.
  cty1-cty&noc $3.
  ctyallemp1-ctyallemp&noc 8.
;
retain
  p1-p&noc
  cty1-cty&noc
ctyemp1-ctyemp&noc
ctyallemp1-ctyallemp&noc
ctyallref1-ctyallref&noc
  ctyi
;
array p{&noc} p1-p&noc;
array cty{&noc} cty1-cty&noc;
array ctyemp{&noc} ctyemp1-ctyemp&noc;
array ctyallemp{&noc} ctyallemp1-ctyallemp&noc;
array ctyallref{&noc} ctyallref1-ctyallref&noc;

if first.es_sic then do i=1 to &noc;
  p[i]=.;
  cty[i]='';
  ctyi=0;
ctyemp(i)=.;
end;
ctyi=ctyi+1;
cty(ctyi)=county;

```

Fill in ctyemp only with all-industry employment. Also fill in the reference array.

```

if es_sic='' then do;
ctyallemp(ctyi)=county_emp;
ctyallref(ctyi)=county;
end;

```

Assign probability $p_{ci} = \frac{E_{ci}}{E_i}$, the ratio of county-employment in industry i , E_{ci} , to state-wide employment in this particular industry, E_i , and fill in the array with the CDF.

```

if es_sic_emp > 0 then do;
  if ctyi=1 then p(ctyi)=county_emp/es_sic_emp;
  else p(ctyi)=p(ctyi-1)+county_emp/es_sic_emp;

```

Find the same county in the all-industry array, and fill in the ctyemp variable..

```

ref=.;
do j = 1 to &noc.;
  if ctyallref(j)=county then ref=j;
  end;
ctyemp(ctyi)=ctyallemp(ref);
end;
else do;
  if ctyi=1 then p(ctyi)=0;
  else p(ctyi)=p(ctyi-1);
end;

  if last.es_sic then output;
run;

/*===== prepare QA =====*/

  data INTERWRK.ecfqa_legprint2;
  set probs(where=(es_sic=''));
run;

/*===== end QA =====*/

```

The dataset IMPUTE does not contain employment. At this point, we use the unit's average employment, as pulled from LEG_STRUCTURE.

```

proc summary data=ecf nway;
  class sein seinunit;
  var best_emp1;
  output out=means mean=mean_unit_emp;
run;

proc sort data=INTERWRK.impute;
by sein seinunit;
run;

  data impute;
merge INTERWRK.impute(in=a)
  means(in=b);
by sein seinunit;
if a;
run;

```

Determine if SIC info is ever available for candidate seinunits;

```

data imp
  (keep=sein seinunit es_sic mean_unit_emp);
set impute;

  length es_sic $2.;
array aes_sic(&leg_qrange) es_sic1-es_sic&leg_qrange;
array algq(&leg_qrange) lgq1-lgq&leg_qrange;

  last_sic='';
do i=1 to &leg_qrange;
  if algq(i) in (10 11) and es_sic='' then es_sic=aes_sic(i);
end;
run;

proc sort data=imp;
  by es_sic;
run;

```

Assign imputed county;

```

data ctyimpute
  fail
  (keep= sein seinunit es_sic)
  ;
merge
  imp(in=in1)
  probs(in=in2);
by es_sic;

if in1 and not in2 then do;
  es_sic='';
  call symput('fail', 'yes');
  output fail;
  end;
if in1 and in2 then do;
  output ctyimpute;
  end;
run;

data ctyimpute
  (keep= sein seinunit es_sic county mean_unit_emp)
interwrk.temp1
  interwrk.temp2
  interwrk.probs_qa
  (keep= sein seinunit es_sic county largest_county p1-p&noc.
largest_prob mean_unit_emp)
  ;
set ctyimpute;
  by es_sic;

array probs(&noc.) p1-p&noc;
array cty(&noc) cty1-cty&noc;
array ctyemp(&noc) ctyemp1-ctyemp&noc;

length largest_county i j drop_prob largest_prob 8;

```

Before assigning counties, eliminate counties for which this unit would constitute more than a `&leg_spike_cutoff` amount of employment. The QA typically becomes worried when employment spikes by 25%.

First, we find the largest county. This county is the default county if all others fail.

```

  largest_county=1;
  do i = 2 to &noc.;
if ctyemp(i)>ctyemp(largest_county) then largest_county=i;
  end;

  output interwrk.temp1;

```

Now we check each county relative to the spike cutoff, and if we find one that fails, we eliminate it from the array, as long as it is not the largest county.

```

  do i = 1 to &noc.;
if
  (mean_unit_emp/(ctyemp(i)/&leg_qrange)>&leg_spike_cutoff./100)
  and i ne largest_county
  then do;

```

Move all items up one in the array.

```

*   put "XXXX ADJUSTING " i= sein= seinunit= _n=
mean_unit_emp= ctyemp(i)= largest_county= ctyemp56=;

    if i<largest_county then largest_county=largest_county-1;

    if i > 1 then drop_prob=probs(i)-probs(i-1);
    else drop_prob=probs(i);

*   if _n_ < 1000 then put " DIAGNOSTIC:" largest_county= i=
es_sic= county= sein= seinunit= mean_unit_emp= p56=;

    do j=i to &noc.-1;
probs(j)=probs(j+1)-drop_prob;
cty(j)=cty(j+1);
ctyemp(j)=ctyemp(j+1);
    end;
*   put "XXXX end of first j loop" sein= seinunit= _n_= p56=;

```

Set the last one to missing

```

probs(&noc.)=. ;
    cty(&noc.)='';
ctyemp(&noc.)=. ;

```

Now adjust all probabilities.

```

do j=1 to &noc.;
    probs(j)=probs(j)/(1-drop_prob);
end;
*   put "XXXX end of second j loop" sein= seinunit= _n_= p56=;
    end; /* end if spike condition */
    end; /* end i loop */

```

This is for QA purposes.

```

largest_prob=max(of p1-p&noc.);

output interwrk.temp2;

```

Now assign the county.

```

z=ranuni(983);
if z le p1 then county=cty1;
else do j=2 to &noc;
    if z gt probs(j-1) and z le probs(j) then county=cty(j);
end;
output ctyimpute;
output interwrk.probs_qa;
run;

```

Prepare QA

```

data interwrk.probs_qa;
  set interwrk.probs_qa;
  array probs(&noc) p1-p&noc;
  do i=1 to &noc;
if round(probs(i),0.000001)=1 then num_counties=i;
end;
  run;

proc print data=temp(obs=1000);
  run;

/*
*   proc print data=fail;
*       title "fail";
*   run;
*/
  data INTERWRK.ecfqa_legfail1;
  set fail;
  run;

/*===== end QA =====*/

```

If we fail to find the county, impute.

```

  %if &fail=yes %then %do;

  data ctyimpute2
    (keep = sein seinunit es_sic county) fail2(keep = sein seinunit
es_sic);
  merge
    fail
      (in=in1)
    probs
      (in=in2 where=(es_sic=""));

  if in1 and in2 then do;
    array probs(&noc) p1-p&noc;
    array cty(&noc) cty1-cty&noc;
    z=ranuni(983);
    if z le p1 then county=cty1;
    else do j=2 to &noc;
      if z gt probs(j-1) and z le probs(j) then county=cty(j);
    end;
    output ctyimpute2;
  end;
  else if in1 and not in2 then
    output fail2;
run;

data ctyimpute;
  set ctyimpute ctyimpute2;
run;

```

Prepare QA

```

/*
*   proc print data=fail2;
*       title "fail2";
*   run;
*/
data INTERWRK.ecfqa_legfail2;
set fail2;
run;

/*
*   proc print data=ctyimpute(obs=1000);
*       title "ctyimpute";
*   run;
*/

/*===== end QA =====*/
%end;

proc sort data=ctyimpute;
by sein seinunit;
run;

```

Combine impute data with the rest of the data;

```

%if ( &test eq ) %then %do;
data INTERWRK.leg_03
(keep= sein seinunit year quarter
es_galid es_geoqual es_geo es_sic
leg_galid leg_geoqual
flag_leggeo leg_geo move t_move mean_unit_emp)
;
merge
INTERWRK.leg_02
ctyimpute(in=in1)
;
by sein seinunit;
if in1 and leg_geoqual in (10, 11) then do;
leg_galid="A"||put(stfips("&state"),z2.)||county||"999999999";
leg_geo=put(stfips("&state"),z2.)||county||"0000000";
end;
run;
%end; /* end of test condition */

```

Prepare QA

```

/*   proc print data=INTERWRK.leg_03 (obs=10 where=(leg_geoqual in (10, 11)));
*       title "LEG obs with imputed county";
*   run;
*
*   proc print data=INTERWRK.leg_03(where=(leg_galid="A&stfips 999999999"));
*       title "ERROR";
*   run;
*
*   proc freq data=INTERWRK.leg_03;
*       title "freq of geoqual";
*       tables leg_geoqual;
*   run;
*/
/*===== end QA =====*/
%mend;

%impute(test=);

```

1.5.2.16 library/sasprogs/16_ecf.sas

```
/* OLDNAME: leg/1.1.04/library/sasprogs/04_leg.sas */  
/* Time-stamp: <05/03/03 19:42:07 vilhuber> */
```

This program takes the dataset from step 3 and merges on additional geocodes from the GAL (to support the QWI and other geography projects), adds labels to all of the variables, and sorts the final dataset so that the key variables can be merged onto the ECF.

```
/*=====*/  
/* 04.01.final.sas */  
/* */  
/* DESCRIPTION */  
/* */  
/* Get additional geo vars, add labels, final sort, etc. */  
/* */  
/* REQUIRES-SAS: config.sas */  
/* */  
/* REQUIRES-DATA: leg.sas7bdat */  
/* REQUIRES-DATA: gal_SS.sas7bdat */  
/* */  
/* PROVIDES-DATA: leg_SS.sas7bdat */  
/*=====*/
```

Sort the data.

```
proc sort data=INTERWRK.leg_03 (rename=(leg_galid=galid))  
  out=leg;  
  by galid;  
run;
```

Final build.

```

%choose_gal2;

data leg_&state
  (keep=sein seinunit year quarter es_galid es_geo
   es_geo_qual es_sic leg_galid leg_subctygeo leg_wib leg_msapmsa
  leg_flag_geo
   leg_geo_qual leg_latitude leg_longitude leg_geocode leg_block
  leg_block_src
   move t_move state
/* variables added per Marc Roemer, 2005-01-26 */
   leg_block_suf1 leg_block_suf2
  )
;
merge
  leg
  (in=in1
   rename=(flag_leggeo=leg_flag_geo leg_geoqual=leg_geo_qual
   es_geoqual=es_geo_qual)
  )
  gal
;
by galid;

length leg_subctygeo $10.;

if in1;
leg_galid=galid;
leg_subctygeo=a_scccc|a_&subcty;
state=put(stfips("&state"),z2.);

label
  es_galid      ="GALID of address on es202"
  es_geo        ="Geocode of address on es202"
  es_geo_qual   ="Quality code for geography from es202"
  es_sic        ="SIC code from es202"
  leg_galid     ="Final GALID"
  leg_subctygeo ="Sub-county geocode"
  leg_wib       ="WIB code, wwwwww"
  leg_msapmsa   ="MSAPMSA metro area code, mmmmmmmmm"
  leg_flag_geo  ="Flag, number of quarters to find geocodes"
  leg_geo_qual  ="Quality of final geography"
  leg_latitude  ="Latitude, 6 implied decimal places"
  leg_longitude ="Longitude, 6 implied decimal places"
  move          ="Flag indicating seinunit move"
  state         ="FIPS State SS"
  t_move        ="Number of time periods since last move"
;
run;

proc sort data=leg_&state. out=OUTPUTS.ecf_&state._leg;
  by sein year quarter seinunit;
run;

```

1.5.2.17 library/sasprogs/17_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/17_ecf.sas >--+ */
/*----- LEGACY NAME: 17_naics_ldb_clean.sas -----*/
/*****
/* REQUIRES-SAS: runtime_ecf.sas (all macros and * %includes are located here)
*/
/* REQUIRES-DATA: ldb_SS_YYYY.sas7bdat (1990:1 to 2001:4 only) */
/*
/* PROVIDES-DATA: naics_ldb_clean_17.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Clean and Standardize LDB NAICS codes";
title2 "Program Location: &curdir./17_naics_ldb_clean.sas";
```

This step is unique to the LDB .

The yearly files must be stacked first .

```
data temp0;
  %set_ldb;
  if naics_ldb=" " then naics_ldb="999999";
run;

%set_ldb_qa;
```

Collapse to the set of records with POTENTIALLY valid industry codes .

```
proc sort data=temp0(keep=sein year quarter seinunit naics_ldb
where=(naics_ldb~="999999")) out=temp1;
  by sein seinunit naics_ldb year quarter;
run;
```

Collapse to the set of unique industry codes for that SEIN SEINUNIT .

```
proc sort data=temp1(keep=sein seinunit naics_ldb) out=temp2 nodupkey;
  by sein seinunit naics_ldb;
run;
```

Clean each industry code . Count the number of times a valid code is found .

```

data temp3(drop=naics_ldb_1997_valid naics_ldb_2002_valid) temp4(keep=sein
seinunit naics_ldb_1997_valid naics_ldb_2002_valid);
  set temp2;
  by sein seinunit;

  length naics_ldb_1997_invalid naics_ldb_1997_valid $1 naics_ldb_1997_format
$7
         naics_ldb_2002_invalid naics_ldb_2002_valid $1 naics_ldb7
naics_ldb_2002_format $7;

  retain naics_ldb_1997_valid naics_ldb_2002_valid;

  /* xxx_valid is whether a valid code is found anywhere during the period */
  /* xxx_invalid is whether a valid code is present in the current period */

  if first.seinunit then do;
    naics_ldb_1997_valid="0";
    naics_ldb_2002_valid="0";
  end;

  naics_ldb7="E" || naics_ldb;
  naics_ldb_1997_invalid="0";
  naics_ldb_2002_invalid="0";

  /* Run NAICS through the Multi-format */
  naics_ldb_1997_format=put(naics_ldb7,$197naic.);
  if substr(naics_ldb_1997_format,1,1)="E" then naics_ldb_1997_invalid="1";

  naics_ldb_2002_format=put(naics_ldb7,$102naic.);
  if substr(naics_ldb_2002_format,1,1)="E" then naics_ldb_2002_invalid="1";

  if naics_ldb_1997_invalid="0" then naics_ldb_1997_valid="1";
  if naics_ldb_2002_invalid="0" then naics_ldb_2002_valid="1";

  output temp3;
  if last.seinunit then output temp4;
run;

proc freq data=temp3;
  tables naics_ldb_1997_invalid naics_ldb_2002_invalid;
run;

proc freq data=temp4;
  tables naics_ldb_1997_valid naics_ldb_2002_valid;
run;

```

Attach the information showing whether a valid code was found . Clean the industry code if no other code is found . Otherwise set to missing . The valid code will be picked up when filling is done .

```

data INTERWRK.temp5_17;
  merge temp3(in=a) temp4(in=b);
  by sein seinunit;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;

  length naics_ldb1997 naics_ldb1997_clean naics_ldb1997_impute $6;

  /* Set up NAICS 1997 variables */
  naics_ldb1997_clean="999999";
  if naics_ldb_1997_invalid="0" then naics_ldb1997_clean=naics_ldb;
  if naics_ldb1997="999999" and naics_ldb_1997_invalid="1" and
naics_ldb_1997_valid="1" then naics_ldb_1997_invalid="6";

  naics_ldb1997=naics_ldb;
  uniform1=ranuni(6386542);

```

For Industry codes not in the multi-format impute using the first X digits .

```

candidate1997=0;
  if naics_ldb1997="999999" and naics_ldb_1997_invalid="1" and
naics_ldb_1997_valid="0" then do;
  candidate1997=1;

```

Check the 5 digit code first .

```

%indlkup(naics1997,naics_ldb1997,5,parms_us_imp_ncs1997_ncs19975,uniform1);
  if naics_ldb1997_clean="999999" then naics_ldb_1997_invalid="5";

```

if not found at 5 digit level check 4 digit .

```

  if naics_ldb1997_clean="999999" then do;

%indlkup(naics1997,naics_ldb1997,4,parms_us_imp_ncs1997_ncs19974,uniform1);
  end;
  if naics_ldb_1997_invalid="1" and naics_ldb1997_clean="999999" then
naics_ldb_1997_invalid="4";

```

if not found at 4 digit level check 3 digit .

```

  if naics_ldb1997_clean="999999" then do;

%indlkup(naics1997,naics_ldb1997,3,parms_us_imp_ncs1997_ncs19973,uniform1);
  end;
  if naics_ldb_1997_invalid="1" and naics_ldb1997_clean="999999" then
naics_ldb_1997_invalid="3";
  end;

```

```

length naics_ldb2002 naics_ldb2002_clean naics_ldb2002_impute $6;

```

```

/* Set up NAICS 2002 variables */
naics_ldb2002_clean="999999";
if naics_ldb_2002_invalid="0" then naics_ldb2002_clean=naics_ldb;
if naics_ldb2002="999999" and naics_ldb_2002_invalid="1" and
naics_ldb_2002_valid="1" then naics_ldb_2002_invalid="6";

naics_ldb2002=naics_ldb;
uniform1=ranuni(6937541);

```

For Industry codes not in the multi-format impute using the first X digits .

```
candidate2002=0;
  if naics_ldb2002~="999999" and naics_ldb_2002_invalid="1" and
naics_ldb_2002_valid="0" then do;
    candidate2002=1;
```

Check the 5 digit code first .

```
%indlkup(naics2002,naics_ldb2002,5,parms_us_imp_ncs2002_ncs20025,uniform1);
  if naics_ldb2002_clean~="999999" then naics_ldb_2002_invalid="5";
```

if not found at 5 digit level check 4 digit .

```
  if naics_ldb2002_clean="999999" then do;

%indlkup(naics2002,naics_ldb2002,4,parms_us_imp_ncs2002_ncs20024,uniform1);
  end;
  if naics_ldb_2002_invalid="1" and naics_ldb2002_clean~="999999" then
naics_ldb_2002_invalid="4";
```

if not found at 4 digit level check 3 digit .

```
  if naics_ldb2002_clean="999999" then do;

%indlkup(naics2002,naics_ldb2002,3,parms_us_imp_ncs2002_ncs20023,uniform1);
  end;
  if naics_ldb_2002_invalid="1" and naics_ldb2002_clean~="999999" then
naics_ldb_2002_invalid="3";
  end;
run;

/** DIAGNOSTIC OUTPUT **/
/* moved to ecfdiag_17.sas */

/* Reattach the cleaned NAICS codes back to the original */

data temp6;
  merge temp1(in=a) INTERWRK.temp5_17(in=b keep=sein seinunit naics_ldb
naics_ldb1997_clean naics_ldb_1997_invalid naics_ldb_1997_valid
naics_ldb_2002_invalid naics_ldb_2002_valid);
  by sein seinunit naics_ldb;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;
run;

/** DIAGNOSTIC OUTPUT **/
/* moved to ecfdiag_17.sas */

proc sort data=temp6 out=INTERWRK.naics_ldb_clean_17;
  by sein year quarter seinunit;
run;
```

1.5.2.18 library/sasprogs/18_ecf.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/18_ecf.sas >--+ */
/*----- LEGACY NAME: 18_seinunit_wide.sas -----*/
```

This program reads in the geography data and transforms the data from long into wide format.

Geography data is brought into the ECF program sequence and checked to ensure the proper record structure is returned. Each GEO variable is also output into a separate file so that the mode value can be calculated in step 11.

Filling missing values requires a transformation of the data into wide format (one record for each SEIN SEINUNIT with the YEAR QUARTER values for a variable stored in arrays). The wide file is used in later programs to make available the complete history for each SEIN SEINUNIT (used to fill missing values).

The following variables are transformed into wide format; SIC, NAICS, county, ownership code, and EIN.

```
/* See runtime_ecf.sas for configuration options */
/* ALL macros are stored in runtime_ecf.sas */

/*****
/* REQUIRES-SAS: runtime_ecf.sas */
/* REQUIRES-DATA: GEO data from /data/master/geo/LEG */
/* REQUIRES-DATA: NAICS data from the BLS LDB */
/* PROVIDES-DATA: ss_employer_char_unit2.sas7bdat */
/* seinunit_county_18.sas7bdat */
/* seinunit_msapmsa_18.sas7bdat */
/* seinunit_state_18.sas7bdat */
/* seinunit_subctygeo_18.sas7bdat */
/* seinunit_wib_18.sas7bdat */
/* seinunit_wide_18.sas7bdat */
/* seinunit_long_18.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Transform data from narrow to wide format";
title2 "Program Location: &curdir./18_seinunit_wide.sas";
```

Bring in the LDB NAICS data .

```

data temp1(drop=_merge) merge_check(keep=year _merge);
merge INTERWRK.&state._employer_char_unit(in=a)
      INTERWRK.naics_ldb_clean_17(in=b keep=sein year quarter seinunit
                                naics_ldb1997_clean naics_ldb_1997_invalid
                                naics_ldb2002_clean naics_ldb_2002_invalid
                                rename=(naics_ldb1997_clean=naics_ldb1997
                                         naics_ldb2002_clean=naics_ldb2002));
      by sein year quarter seinunit;

if a=1 and b=1 then _merge=3;
if a=1 and b=0 then _merge=1;
if a=0 and b=1 then _merge=2;

if naics_ldb1997=" " then naics_ldb1997="999999";
if naics_ldb2002=" " then naics_ldb2002="999999";

if naics_ldb_1997_invalid=" " then naics_ldb_1997_invalid="1";
if naics_ldb_2002_invalid=" " then naics_ldb_2002_invalid="1";

if a=1 then output temp1;
output merge_check;
run;

proc freq data=merge_check;
title "Merge Check of LDB NAICS";
tables year*_merge;
run;

```

Bring in the GEO data .

```

data INTERWRK.&state._employer_char_unit2;
merge temp1(in=a)
      OUTPUTS.ecf.&state._leg(in=b keep=sein year quarter seinunit
                              leg_latitude
                              leg_longitude leg_subctygeo leg_geo_qual
                              leg_flag_geo leg_wib leg_msapmsa es_galid
                              leg_galid);
      by sein year quarter seinunit;

if a=1 and b=1 then _merge=3;
if a=1 and b=0 then _merge=1;
if a=0 and b=1 then _merge=2;

```

Create the LEG county and state variables .

```

length leg_state es_state $2 leg_county $3;

es_state="&stfips";

leg_state=substr(leg_subctygeo,1,2);
leg_county=substr(leg_subctygeo,3,3);
run;

/*---- START QA PREP----*/

proc contents;
run;

proc freq;
title "Merge Check of GEO Data";
tables year*_merge leg_state es_state leg_county;
run;
/*---- END QA PREP----*/

```

Sort the seinunit level data .

```
proc sort data=INTERWRK.&state._employer_char_unit2
  (keep=sein year quarter seinunit yr_qtr sic naics1997 naics2002
    naics_aux1997 naics_aux2002 naics_ldb1997 naics_ldb2002
    county owner_code ein ein_defect
    best_emp1 best_emp2 best_emp3
    leg_wib leg_msapmsa leg_state leg_county leg_subctygeo es_galid
  leg_galid
  )
  out=step1;
  by sein seinunit year quarter;
run;
```

Create the wide dataset .

```
data INTERWRK.seinunit_wide_18(
  keep=sein seinunit
  %bldstr(sic,no)
  %bldstr(naics1997,no)
  %bldstr(naics2002,no)
  %bldstr(naics_aux1997,no)
  %bldstr(naics_aux2002,no)
  %bldstr(naics_ldb1997,no)
  %bldstr(naics_ldb2002,no)
  %bldstr(fips,no)
  %bldstr(en,yes)
  %bldstr(owner_code,no)
)
  INTERWRK.seinunit_long_18(keep=sein year quarter seinunit yr_qtr best_emp1
best_emp2 best_emp3)
  INTERWRK.seinunit_wib_18(keep=sein year quarter seinunit leg_wib best_emp1
best_emp2 best_emp3)
  INTERWRK.seinunit_msapmsa_18(keep=sein year quarter seinunit leg_msapmsa
best_emp1 best_emp2 best_emp3)
  INTERWRK.seinunit_state_18(keep=sein year quarter seinunit leg_state
best_emp1 best_emp2 best_emp3)
  INTERWRK.seinunit_county_18(keep=sein year quarter seinunit leg_county
best_emp1 best_emp2 best_emp3)
  INTERWRK.seinunit_subctygeo_18(keep=sein year quarter seinunit
leg_subctygeo best_emp1 best_emp2 best_emp3)
;

  set step1;
  by sein seinunit;
```

Set the default length of each variable in the array .

```

/* SIC */
length %bldstr(sic,no) $4;

/* NAICS */
length %bldstr(naics1997,no) $6;
length %bldstr(naics2002,no) $6;
length %bldstr(naics_aux1997,no) $6;
length %bldstr(naics_aux2002,no) $6;
length %bldstr(naics_ldb1997,no) $6;
length %bldstr(naics_ldb2002,no) $6;

/* COUNTY */
length %bldstr(fips,no) $3;

/* OWNER_CODE */
length %bldstr(owner_code,no) $1;

/* EIN */
length %bldstr(en,yes) $9;

```

Retain the Variables in the array .

```

retain
  %bldstr(sic,no)
  %bldstr(naics1997,no)
  %bldstr(naics2002,no)
  %bldstr(naics_aux1997,no)
  %bldstr(naics_aux2002,no)
  %bldstr(naics_ldb1997,no)
  %bldstr(naics_ldb2002,no)
  %bldstr(fips,no)
  %bldstr(en,yes)
  %bldstr(owner_code,no)
;

```

Set up the arrays .

```

array sc{&start_year:&end_year,1:4} %bldstr(sic,no);

array ncs97{&start_year:&end_year,1:4} %bldstr(naics1997,no);
array ncs02{&start_year:&end_year,1:4} %bldstr(naics2002,no);
array ncx97{&start_year:&end_year,1:4} %bldstr(naics_aux1997,no);
array ncx02{&start_year:&end_year,1:4} %bldstr(naics_aux2002,no);
array ncl97{&start_year:&end_year,1:4} %bldstr(naics_ldb1997,no);
array ncl02{&start_year:&end_year,1:4} %bldstr(naics_ldb2002,no);

array owcd{&start_year:&end_year,1:4} %bldstr(owner_code,no);

array fip{&start_year:&end_year,1:4} %bldstr(fips,no);

array en{&start_year:&end_year,1:4} %bldstr(en,yes);

```

Load the arrays .

```

if first.seinunit then do;
  do i=&start_year to &end_year;
    do j=1 to 4;
      sc{i,j}="9999";
      ncs97{i,j}="999999";
      ncs02{i,j}="999999";
      ncx97{i,j}="999999";
      ncx02{i,j}="999999";
      ncl97{i,j}="999999";
      ncl02{i,j}="999999";
      owcd{i,j}="9";
      fip{i,j}="ZZZ";
      en{i,j}=" ";
    end;
  end;
end;

sc{year,quarter}=sic;
ncs97{year,quarter}=naics1997;
ncs02{year,quarter}=naics2002;
ncx97{year,quarter}=naics_aux1997;
ncx02{year,quarter}=naics_aux2002;
ncl97{year,quarter}=naics_ldb1997;
ncl02{year,quarter}=naics_ldb2002;
owcd{year,quarter}=owner_code;
fip{year,quarter}=county;
if ein_defect=0 then en{year,quarter}=ein;

if last.seinunit then output INTERWRK.seinunit_wide_18;
output INTERWRK.seinunit_long_18;
output INTERWRK.seinunit_wib_18;
output INTERWRK.seinunit_msapmsa_18;
output INTERWRK.seinunit_state_18;
output INTERWRK.seinunit_county_18;
output INTERWRK.seinunit_subctygeo_18;
run;

/* diagnostics moved to ecfdiag_18.sas */

```

1.5.2.19 library/sasprogs/19_ecf.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/19_ecf.sas >--+ */
/*----- LEGACY NAME: 19_seinunit_fill.sas -----*/

```

Missing values are filled and NAICS values are used to impute SIC.

The variables in program 18 that were transformed into wide format are filled, if missing, with the closest value from another YEAR QUARTER.

```

/* See runtime_ecf.sas for configuration options */
/* ALL macros are stored in runtime_ecf.sas */

/*****
/* REQUIRES-SAS: runtime_ecf.sas
/* REQUIRES-DATA: seinunit_long_18.sas7bdat, seinunit_wide_18.sas7bdat
/* PROVIDES-DATA: seinunit_long_19.sas7bdat
*****/

* %include "runtime_ecf.sas";

title1 "Put Everything together";
title2 "Program Location: &curdir./19_fill_sync.sas";

```

Begin the fill .

```

data INTERWRK.seinunit_long_19
  (keep=sein year quarter seinunit yr_qtr _merge
    best_emp1-best_emp3
    es_sic es_sic_miss es_sic_flag
    es_naics1997 es_naics1997_miss es_naics1997_flag
    es_naics2002 es_naics2002_miss es_naics2002_flag
    es_naics_aux1997 es_naics_aux1997_miss es_naics_aux1997_flag
    es_naics_aux2002 es_naics_aux2002_miss es_naics_aux2002_flag
    es_naics_ldb1997 es_naics_ldb1997_miss es_naics_ldb1997_flag
    es_naics_ldb2002 es_naics_ldb2002_miss es_naics_ldb2002_flag
    es_county es_county_miss es_county_flag
    es_owner_code es_owner_code_miss es_owner_code_flag
    es_ein es_ein_miss es_ein_flag
  );

  merge INTERWRK.seinunit_long_18(in=_mvar1)
  INTERWRK.seinunit_wide_18(in=_mvar2);
  by sein seinunit;

  length ES_SIC $4 ES_COUNTY $3 ES_EIN $9 ES_NAICS1997 ES_NAICS2002
  ES_NAICS_AUX1997 ES_NAICS_AUX2002
  ES_NAICS_LDB1997 ES_NAICS_LDB2002 $6 ES_OWNER_CODE $1;

  if _mvar1=1 and _mvar2=1 then _merge=3;
  if _mvar1=1 and _mvar2=0 then _merge=1;
  if _mvar1=0 and _mvar2=1 then _merge=2;

```

Load the Data into Arrays .

```

array sc{&start_year:&end_year,1:4} %bldstr(sic,no);

array ncs97{&start_year:&end_year,1:4} %bldstr(naics1997,no);
array ncs02{&start_year:&end_year,1:4} %bldstr(naics2002,no);
array ncx97{&start_year:&end_year,1:4} %bldstr(naics_aux1997,no);
array ncx02{&start_year:&end_year,1:4} %bldstr(naics_aux2002,no);
array ncl97{&start_year:&end_year,1:4} %bldstr(naics_ldb1997,no);
array ncl02{&start_year:&end_year,1:4} %bldstr(naics_ldb2002,no);

array owcd{&start_year:&end_year,1:4} %bldstr(owner_code,no);

array fip{&start_year:&end_year,1:4} %bldstr(fips,no);

array en{&start_year:&end_year,1:4} %bldstr(en,yes);

```

Assign the data to the appropriate year .

```

ES_SIC=sc{year,quarter};
ES_NAICS1997=ncs97{year,quarter};
ES_NAICS2002=ncs02{year,quarter};
ES_NAICS_AUX1997=ncx97{year,quarter};
ES_NAICS_AUX2002=ncx02{year,quarter};
ES_NAICS_LDB1997=nc197{year,quarter};
ES_NAICS_LDB2002=nc102{year,quarter};
ES_COUNTY=fip{year,quarter};
ES_OWNER_CODE=owcd{year,quarter};
ES_EIN=en{year,quarter};

```

```

/*****

```

Process Variables to fill.

Set flag if data is missing .

```

es_sic_miss=0;
es_naics1997_miss=0;
es_naics2002_miss=0;
es_naics_aux1997_miss=0;
es_naics_aux2002_miss=0;
es_naics_ldb1997_miss=0;
es_naics_ldb2002_miss=0;
es_county_miss=0;
es_owner_code_miss=0;
es_ein_miss=0;
if es_sic='9999' then do;
    es_sic_miss=1;
end;
if es_naics1997='999999' then do;
    es_naics1997_miss=1;
end;
if es_naics2002='999999' then do;
    es_naics2002_miss=1;
end;
if es_naics_aux1997='999999' then do;
    es_naics_aux1997_miss=1;
end;
if es_naics_aux2002='999999' then do;
    es_naics_aux2002_miss=1;
end;
if es_naics_ldb1997='999999' then do;
    es_naics_ldb1997_miss=1;
end;
if es_naics_ldb2002='999999' then do;
    es_naics_ldb2002_miss=1;
end;
if es_county='ZZZ' then do;
    es_county_miss=1;
end;
if es_owner_code='9' then do;
    es_owner_code_miss=1;
end;
if es_ein=" " then do;
    es_ein_miss=1;
end;

```

ES_SIC cleanup

```

if es_sic_miss=1 then do;
    %qloop(es_sic,sc,%str('9999'));
end;
else do;
    es_sic_flag=0;
end;
if es_sic_miss=1 and es_sic~="9999" then es_sic_miss=2;

```

ES_NAICS1997 cleanup .

```

if es_naics1997_miss=1 then do;
    %qloop(es_naics1997,ncs97,%str('999999'));
end;
else do;
    es_naics1997_flag=0;
end;
if es_naics1997_miss=1 and es_naics1997~="999999" then
es_naics1997_miss=2;

```

ES_NAICS2002 cleanup.

```

if es_naics2002_miss=1 then do;
    %qloop(es_naics2002,ncs02,%str('999999'));
end;
else do;
    es_naics2002_flag=0;
end;
if es_naics2002_miss=1 and es_naics2002~="999999" then
es_naics2002_miss=2;

```

ES_NAICS_AUX1997 cleanup.

```

if es_naics_aux1997_miss=1 then do;
    %qloop(es_naics_aux1997,ncx97,%str('999999'));
end;
else do;
    es_naics_aux1997_flag=0;
end;
if es_naics_aux1997_miss=1 and es_naics_aux1997~="999999" then
es_naics_aux1997_miss=2;

```

ES_NAICS_AUX2002 cleanup .

```

if es_naics_aux2002_miss=1 then do;
    %qloop(es_naics_aux2002,ncx02,%str('999999'));
end;
else do;
    es_naics_aux2002_flag=0;
end;
if es_naics_aux2002_miss=1 and es_naics_aux2002~="999999" then
es_naics_aux2002_miss=2;

```

ES_NAICS_LDB1997 cleanup.

```

if es_naics_ldb1997_miss=1 then do;
    %qloop(es_naics_ldb1997,nc197,%str('999999'));
end;
else do;
    es_naics_ldb1997_flag=0;
end;
if es_naics_ldb1997_miss=1 and es_naics_ldb1997~="999999" then
es_naics_ldb1997_miss=2;

```

ES_NAICS_LDB2002 cleanup.

```

if es_naics_ldb2002_miss=1 then do;
    %qloop(es_naics_ldb2002,nc102,%str('999999'));
end;
else do;
    es_naics_ldb2002_flag=0;
end;
if es_naics_ldb2002_miss=1 and es_naics_ldb2002~="999999" then
es_naics_ldb2002_miss=2;

```

ES_COUNTY cleanup.

```

if es_county_miss=1 then do;
    %qloop(es_county,fip,%str('ZZZ'));
end;
else do;
    es_county_flag=0;
end;
if es_county_miss=1 and es_county~="ZZZ" then es_county_miss=2;

```

ES_OWNER_CODE cleanup .

```

if es_owner_code_miss=1 then do;
    %qloop(es_owner_code,owcd,%str('9'));
end;
else do;
    es_owner_code_flag=0;
end;
if es_owner_code_miss=1 and es_owner_code~="9" then es_owner_code_miss=2;

```

EIN cleanup.

```

if es_ein_miss=1 then do;
%macro ckkein;
    %if &einavail=ein %then %do;
        %qloop(es_ein,en,%str(' '));
    %end;
%mend;
%ckkein;
end;
else do;
    es_ein_flag=0;
end;
if es_ein_miss=1 and es_ein~=" " then es_ein_miss=2;

```

run;

Diagnostics moved to ecfdiag_19.sas

1.5.2.20 library/sasprogs/20_ecf.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/20_ecf.sas >--+ */
/*----- LEGACY NAME: 20_seinunit_sync.sas -----*/

```

KEVIN: brief description?

```

/*****
/* REQUIRES-DATA: seinunit_long_19.sas7bdat */
/* PROVIDES-DATA: seinunit_county_20.sas7bdat */
/* seinunit_ein_20.sas7bdat */
/* seinunit_naics_20.sas7bdat */
/* seinunit_owner_code_20.sas7bdat */
/* seinunit_sic_20.sas7bdat */
/* seinunit_long_20.sas7bdat */
*****/

title1 "Put Everything together";
title2 "Program Location: &curdir./20_fill_sync.sas";

```

Find out which SEINUNITs have at least one valid industry value.

```

data temp1
  (keep=sein seinunit es_sic_valid es_naics1997_valid es_naics2002_valid
    es_naics_aux1997_valid es_naics_aux2002_valid es_naics_ldb1997_valid
    es_naics_ldb2002_valid any_valid sic_change
  );
  set INTERWRK.seinunit_long_19();
  by sein seinunit;

  retain es_sic_valid es_naics1997_valid es_naics2002_valid
    es_naics_aux1997_valid es_naics_aux2002_valid
    es_naics_ldb1997_valid es_naics_ldb2002_valid
    sic_change es_sic_last;

  length es_sic_last $4;

  if first.seinunit then do;
    es_sic_valid=0;
    es_naics1997_valid=0;
    es_naics2002_valid =0;
    es_naics_aux1997_valid=0;
    es_naics_aux2002_valid=0;
    es_naics_ldb1997_valid=0;
    es_naics_ldb2002_valid=0;
    sic_change=.;
    es_sic_last=" ";
  end;

```

Check for valid industry values

```

  if es_sic~="9999" and es_sic~=" " then es_sic_valid=1;
  if es_naics1997~="9999999" and es_naics1997~=" " then es_naics1997_valid=1;
  if es_naics2002~="9999999" and es_naics2002~=" " then es_naics2002_valid=1;
  if es_naics_aux1997~="9999999" and es_naics_aux1997~=" " then
es_naics_aux1997_valid=1;
  if es_naics_aux2002~="9999999" and es_naics_aux2002~=" " then
es_naics_aux2002_valid=1;
  if es_naics_ldb1997~="9999999" and es_naics_ldb1997~=" " then
es_naics_ldb1997_valid=1;
  if es_naics_ldb2002~="9999999" and es_naics_ldb2002~=" " then
es_naics_ldb2002_valid=1;

```

Look for SIC changes for 1998 and earlier

```

if sic_change=. and es_sic~="9999" then sic_change=0;

if year<2000 and sic_change>. and es_sic_last~=" " and es_sic_last~=es_sic
then sic_change=sic_change+1;

es_sic_last=es_sic;

if last.seinunit then do;
  any_valid=0;
  if es_sic_valid=1 or es_naics1997_valid=1 or es_naics2002_valid=1 or
es_naics_aux1997_valid=1
    or es_naics_aux2002_valid=1 or es_naics_ldb1997_valid=1 or
es_naics_ldb2002_valid=1 then any_valid=1;
  output;
end;
run;

```

Diagnostics

```

proc freq data=temp1;
  tables es_sic_valid es_naics1997_valid es_naics2002_valid
  es_naics_aux1997_valid es_naics_aux2002_valid
  es_naics_ldb1997_valid es_naics_ldb2002_valid
  any_valid sic_change;
run;

proc print data=temp1(obs=100);
run;

```

Begin the fill.

```

data INTERWRK.seinunit_long_20
  INTERWRK.seinunit_sic_20
    (keep=sein year quarter seinunit best_emp1-best_emp3 es_sic)
  INTERWRK.seinunit_naics_eso1997_20
    (keep=sein year quarter seinunit best_emp1-best_emp3 es_naics_eso1997)
  INTERWRK.seinunit_naics_eso2002_20
    (keep=sein year quarter seinunit best_emp1-best_emp3 es_naics_eso2002)
  INTERWRK.seinunit_naics_fnl1997_20
    (keep=sein year quarter seinunit best_emp1-best_emp3 es_naics_fnl1997)
  INTERWRK.seinunit_naics_fnl2002_20
    (keep=sein year quarter seinunit best_emp1-best_emp3 es_naics_fnl2002)
  INTERWRK.seinunit_county_20
    (keep=sein year quarter seinunit best_emp1-best_emp3 es_county)
  INTERWRK.seinunit_owner_code_20
    (keep=sein year quarter seinunit best_emp1-best_emp3 es_owner_code)
  INTERWRK.seinunit_ein_20
    (keep=sein year quarter seinunit best_emp1-best_emp3 es_ein);

merge INTERWRK.seinunit_long_19(in=a ) temp1(in=b);
  by sein seinunit;

if a=1 and b=1 then merge=3;
if a=1 and b=0 then merge=1;
if a=0 and b=1 then merge=2;

length ES_NAICS_FNL1997 ES_NAICS_FNL2002
  ES_NAICS_ESO1997 ES_NAICS_ESO2002
  ES_NAICS_IMP1997 ES_NAICS_IMP2002 $6
  ES_SIC_SRC ES_NAICS1997_SRC ES_NAICS2002_SRC
  ES_NAICS_AUX1997_SRC ES_NAICS_AUX2002_SRC
  ES_NAICS_LDB1997_SRC ES_NAICS_LDB2002_SRC
  ES_NAICS_FNL1997_SRC ES_NAICS_FNL2002_SRC
  ES_NAICS_IMP1997_SRC ES_NAICS_IMP2002_SRC
  ES_NAICS_ESO1997_SRC ES_NAICS_ESO2002_SRC $3;

retain uniform0 uniform1 uniform2 uniform3;

```

We want the imputed value to be the same each period We also want each NAICS variable to get the same impute for a given input

```

if first.sein then do;
  uniform0=uniform(49872232);
  uniform1=uniform(23938577);
  uniform2=uniform(77643431);
  uniform3=uniform(98231454);
end;

```

SYNC the NAICS VARIABLE

```

es_naics1997_src=" ";
es_naics2002_src=" ";
if es_naics1997_miss=0 or es_naics1997_miss=2 then es_naics1997_src="N97";
if es_naics2002_miss=0 or es_naics2002_miss=2 then es_naics2002_src="N02";

```

Fill NAICS1997 Missings using NAICS2002.

```

%indlkup2(es_naics1997,es_naics2002,6,6,es_naics2002,naics1997_impute,uniform0,p
arms_us_imp_ncs1997_ncs2002);
  if lookup_temp=1 then es_naics1997_src="N02";

```

Fill NAICS1997 Missings using SIC.

```

%indlkup2(es_naics1997,es_sic,6,4,es_sic,naics1997_impute,uniform1,parms_us_imp_
ncs1997_sic);
  if lookup_temp=1 then es_naics1997_src="SIC";
  if lookup_temp=2 and found_ind_temp=0 then do;
    es_naics1997=es_naics1997_clean;
    es_naics1997_miss=1.5;
    es_naics1997_src="SIC";
  end;

```

Fill NAICS2002 Missings using NAICS1997

```

  miss_naics1997_temp=es_naics1997_miss;
  if es_naics1997_miss=2 or es_naics1997_miss=3 or es_naics1997_miss=4 then
es_naics1997_miss=0;

%indlkup2(es_naics2002,es_naics1997,6,6,es_naics1997,naics2002_impute,uniform2,p
arms_us_imp_ncs2002_ncs1997);
  if lookup_temp=1 then es_naics2002_src=es_naics1997_src;
  if lookup_temp=2 and found_ind_temp=0 and es_naics1997_miss=1.5 then do;
    es_naics2002=es_naics2002_clean;
    es_naics2002_miss=1.5;
    es_naics2002_src=es_naics1997_src;
  end;
  es_naics1997_miss=miss_naics1997_temp;

```

SYNC the NAICS_AUX VARIABLE

```

  es_naics_aux1997_src=" ";
  es_naics_aux2002_src=" ";
  if es_naics_aux1997_miss=0 or es_naics_aux1997_miss=2 then
es_naics_aux1997_src="N97";
  if es_naics_aux2002_miss=0 or es_naics_aux2002_miss=2 then
es_naics_aux2002_src="N02";

```

Fill NAICS_AUX1997 Missings using NAICS_AUX2002.

```

  es_naics2002_temp=es_naics2002;

%indlkup2(es_naics_aux1997,es_naics_aux2002,6,6,es_naics2002,naics1997_impute,un
iform0,parms_us_imp_ncs1997_ncs2002);
  if lookup_temp=1 then es_naics_aux1997_src="N02";
  es_naics2002=es_naics2002_temp;

```

Fill NAICS_AUX1997 Missings using SIC

```

%indlkup2(es_naics_aux1997,es_sic,6,4,es_sic,naics1997_impute,uniform1,parms_us_
imp_ncs1997_sic);
  if lookup_temp=1 then es_naics_aux1997_src="SIC";
  if lookup_temp=2 and found_ind_temp=0 then do;
    es_naics_aux1997=es_naics_aux1997_clean;
    es_naics_aux1997_miss=1.5;
    es_naics_aux1997_src="SIC";
  end;

```

Fill NAICS_AUX2002 Missings using NAICS_AUX1997.

```

es_naics1997_temp=es_naics1997;
miss_naics_aux1997_temp=es_naics_aux1997_miss;
if es_naics_aux1997_miss=2 or es_naics_aux1997_miss=3 or
es_naics_aux1997_miss=4 then es_naics_aux1997_miss=0;

%indlkup2(es_naics_aux2002,es_naics_aux1997,6,6,es_naics1997,naics2002_impute,un
iform2,parms_us_imp_ncs2002_ncs1997);
if lookup_temp=1 then es_naics_aux2002_src=es_naics_aux1997_src;
if lookup_temp=2 and found_ind_temp=0 and es_naics_aux1997_miss=1.5 then
do;
    es_naics_aux2002=es_naics_aux2002_clean;
    es_naics_aux2002_miss=1.5;
    es_naics_aux2002_src=es_naics_aux1997_src;
end;
es_naics_aux1997_miss=miss_naics_aux1997_temp;
es_naics1997=es_naics1997_temp;

```

SYNC the NAICS_LDB VARIABLE.

```

es_naics_ldb1997_src=" ";
es_naics_ldb2002_src=" ";
if es_naics_ldb1997_miss=0 or es_naics_ldb1997_miss=2 then
es_naics_ldb1997_src="N97";
if es_naics_ldb2002_miss=0 or es_naics_ldb2002_miss=2 then
es_naics_ldb2002_src="N02";

```

Fill NAICS_LDB1997 Missings using NAICS_LDB2002.

```

es_naics2002_temp=es_naics2002;

%indlkup2(es_naics_ldb1997,es_naics_ldb2002,6,6,es_naics2002,naics1997_impute,un
iform0,parms_us_imp_ncs1997_ncs2002);
if lookup_temp=1 then es_naics_ldb1997_src="N02";
es_naics2002=es_naics2002_temp;

```

Fill NAICS_LDB1997 Missings using SIC.

```

%indlkup2(es_naics_ldb1997,es_sic,6,4,es_sic,naics1997_impute,uniform1,parms_us_
imp_ncs1997_sic);
if lookup_temp=1 then es_naics_ldb1997_src="SIC";
if lookup_temp=2 and found_ind_temp=0 then do;
    es_naics_ldb1997=es_naics_ldb1997_clean;
    es_naics_ldb1997_miss=1.5;
    es_naics_ldb1997_src="SIC";
end;

```

Fill NAICS_LDB2002 Missings using NAICS_LDB1997.

```

es_naics1997_temp=es_naics1997;
miss_naics_ldb1997_temp=es_naics_ldb1997_miss;
if es_naics_ldb1997_miss=2 or es_naics_ldb1997_miss=3 or
es_naics_ldb1997_miss=4 then es_naics_ldb1997_miss=0;

%indlkup2(es_naics_ldb2002,es_naics_ldb1997,6,6,es_naics1997,naics2002_impute,un
iform2,parms_us_imp_ncs2002_ncs1997);
if lookup_temp=1 then es_naics_ldb2002_src=es_naics_ldb1997_src;
if lookup_temp=2 and found_ind_temp=0 and es_naics_ldb1997_miss=1.5 then
do;
    es_naics_ldb2002=es_naics_ldb2002_clean;
    es_naics_ldb2002_miss=1.5;
    es_naics_ldb2002_src=es_naics_ldb1997_src;
end;
es_naics_ldb1997_miss=miss_naics_ldb1997_temp;
es_naics1997=es_naics1997_temp;

```

Create a unified NAICS variable using only ES202 info.

```
es_naics_eso1997="999999";
es_naics_eso1997_miss=1;
es_naics_eso1997_src=" ";
if es_naics_aux1997~="999999" and es_naics1997~="999999" then do;
  if es_naics_aux1997_miss<=es_naics1997_miss then do;
    es_naics_eso1997=es_naics_aux1997;
    es_naics_eso1997_miss=es_naics_aux1997_miss;
    es_naics_eso1997_src="AUX";
  end;
  else do;
    es_naics_eso1997=es_naics1997;
    es_naics_eso1997_miss=es_naics1997_miss;
    es_naics_eso1997_src="NCS";
  end;
end;
else if es_naics_aux1997~="999999" then do;
  es_naics_eso1997=es_naics_aux1997;
  es_naics_eso1997_miss=es_naics_aux1997_miss;
  es_naics_eso1997_src="AUX";
end;
else if es_naics1997~="999999" then do;
  es_naics_eso1997=es_naics1997;
  es_naics_eso1997_miss=es_naics1997_miss;
  es_naics_eso1997_src="NCS";
end;

es_naics_eso2002="999999";
es_naics_eso2002_miss=1;
es_naics_eso2002_src=" ";
if es_naics_aux2002~="999999" and es_naics2002~="999999" then do;
  if es_naics_aux2002_miss<=es_naics2002_miss then do;
    es_naics_eso2002=es_naics_aux2002;
    es_naics_eso2002_miss=es_naics_aux2002_miss;
    es_naics_eso2002_src="AUX";
  end;
  else do;
    es_naics_eso2002=es_naics2002;
    es_naics_eso2002_miss=es_naics2002_miss;
    es_naics_eso2002_src="NCS";
  end;
end;
else if es_naics_aux2002~="999999" then do;
  es_naics_eso2002=es_naics_aux2002;
  es_naics_eso2002_miss=es_naics_aux2002_miss;
  es_naics_eso2002_src="AUX";
end;
else if es_naics2002~="999999" then do;
  es_naics_eso2002=es_naics2002;
  es_naics_eso2002_miss=es_naics2002_miss;
  es_naics_eso2002_src="NCS";
end;
end;
```

Create a unified NAICS variable using LDB and ES202 info.

```

es_naics_fnl1997="999999";
es_naics_fnl1997_miss=1;
es_naics_fnl1997_src=" ";
if es_naics_ldb1997~="999999" and es_naics_eso1997~="999999" then do;
  if es_naics_ldb1997_miss<=es_naics_eso1997_miss then do;
    es_naics_fnl1997=es_naics_ldb1997;
    es_naics_fnl1997_miss=es_naics_ldb1997_miss;
    es_naics_fnl1997_src="LDB";
  end;
else do;
  es_naics_fnl1997=es_naics_eso1997;
  es_naics_fnl1997_miss=es_naics_eso1997_miss;
  es_naics_fnl1997_src=es_naics_eso1997_src;
end;
end;
else if es_naics_ldb1997~="999999" then do;
  es_naics_fnl1997=es_naics_ldb1997;
  es_naics_fnl1997_miss=es_naics_ldb1997_miss;
  es_naics_fnl1997_src="LDB";
end;
else if es_naics_eso1997~="999999" then do;
  es_naics_fnl1997=es_naics_eso1997;
  es_naics_fnl1997_miss=es_naics_eso1997_miss;
  es_naics_fnl1997_src=es_naics_eso1997_src;
end;

es_naics_fnl2002="999999";
es_naics_fnl2002_miss=1;
es_naics_fnl2002_src=" ";
if es_naics_ldb2002~="999999" and es_naics_eso2002~="999999" then do;
  if es_naics_ldb2002_miss<=es_naics_eso2002_miss then do;
    es_naics_fnl2002=es_naics_ldb2002;
    es_naics_fnl2002_miss=es_naics_ldb2002_miss;
    es_naics_fnl2002_src="LDB";
  end;
else do;
  es_naics_fnl2002=es_naics_eso2002;
  es_naics_fnl2002_miss=es_naics_eso2002_miss;
  es_naics_fnl2002_src=es_naics_eso2002_src;
end;
end;
else if es_naics_ldb2002~="999999" then do;
  es_naics_fnl2002=es_naics_ldb2002;
  es_naics_fnl2002_miss=es_naics_ldb2002_miss;
  es_naics_fnl2002_src="LDB";
end;
else if es_naics_eso2002~="999999" then do;
  es_naics_fnl2002=es_naics_eso2002;
  es_naics_fnl2002_miss=es_naics_eso2002_miss;
  es_naics_fnl2002_src=es_naics_eso2002_src;
end;
end;

```

Fill SIC Missings using NAICS_FNL1997.

```

es_naics1997_temp=es_naics1997;
es_sic_src=" ";
if es_sic_miss=0 or es_sic_miss=2 then es_sic_src="SIC";

%indlkup2(es_sic,es_naics_fnl1997,4,6,es_naics1997,sic_impute,uniform3,parms_us_
imp_sic_ncs1997);
if lookup_temp=1 then es_sic_src="N97";
es_naics1997=es_naics1997_temp;

```

Impute NAICS1997 using only SIC.

```

es_naics_imp1997="999999";
es_naics_imp1997_miss=1;
es_naics_imp1997_src=" ";

%indlkup2(es_naics_imp1997,es_sic,6,4,es_sic,naics1997_impute,uniform1,parms_us_
imp_ncs1997_sic);
    if lookup_temp=1 then es_naics_imp1997_src=es_sic_src;

```

Impute NAICS2002 using only NAICS1997 .

```

es_naics1997_temp=es_naics1997;
es_naics_imp2002="999999";
es_naics_imp2002_miss=1;
es_naics_imp2002_src=" ";

%indlkup2(es_naics_imp2002,es_naics_imp1997,6,6,es_naics1997,naics2002_impute,un
iform2,parms_us_imp_ncs2002_ncs1997);
    if lookup_temp=1 then es_naics_imp2002_src=es_naics_imp1997_src;
    es_naics1997=es_naics1997_temp;

*if es_naics1997=" " then put sein seinunit year quarter;
run;

```

Diagnostics moved to ecfdiag_20.sas

```

proc sort data=INTERWRK.seinunit_long_20;
    by sein year quarter seinunit;
run;

```

1.5.2.21 library/sasprogs/21_ecf.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/21_ecf.sas >--+ */
/*----- LEGACY NAME: 21_sein_mode_calc.sas -----*/

```

This program calculates the modal SIC, NAICS, and county for each SEIN YEAR QUARTER.

In programs 18 and 20, each variable that will have the mode calculated was placed into a separate file along with a measure of employment. The resulting datasets are relatively narrow and thus less costly to sort than the complete dataset (each dataset or variable requires a different sort order).

Using SIC as an example, the data is first sorted by SEIN YEAR QUARTER SIC. The number of SEINUNIT's or employment is calculated for each set of subunits that are in the same industry. The SIC code with the most SEINUNIT's or EMPLOYMENT is the mode.

The unit and employment weighted modal values are calculated over a different universe since not every SEINUNIT in the es202 has positive employment. Thus, the unit weighted mode is still available even when the firm is inactive.

The modal values for each SEIN YEAR QUARTER are merged together to create one dataset containing all the mode variables.

```

/*****
/* REQUIRES-DATA: seinunit_county_18.sas7bdat */
/*                seinunit_msapmsa_18.sas7bdat */
/*                seinunit_state_18.sas7bdat */
/*                seinunit_subctygeo_18.sas7bdat */
/*                seinunit_wib_18.sas7bdat */
/*                seinunit_ein_20.sas7bdat */
/*                seinunit_county_20.sas7bdat */
/*                seinunit_naics_eso1997_20.sas7bdat */
/*                seinunit_naics_eso2002_20.sas7bdat */
/*                seinunit_naics_fnl1997_20.sas7bdat */
/*                seinunit_naics_fnl2002_20.sas7bdat */
/*                seinunit_owner_code_20.sas7bdat */
/*                seinunit_sic_20.sas7bdat */
/* PROVIDES-DATA: sein_modes_21.sas7bdat */
*****/

* %include "runtime_ecf.sas";

title1 "Modal SIC, NAICS, County, Owner_code, EIN";
title2 "Program Location: &curdir./11_sein_mode_calc.sas";

```

Sort the data by each mode var .

```

proc sort data=INTERWRK.seinunit_sic_20;
  by sein year quarter es_sic;
run;

proc sort data=INTERWRK.seinunit_naics_eso1997_20;
  by sein year quarter es_naics_eso1997;
run;

proc sort data=INTERWRK.seinunit_naics_eso2002_20;
  by sein year quarter es_naics_eso2002;
run;

proc sort data=INTERWRK.seinunit_naics_fnl1997_20;
  by sein year quarter es_naics_fnl1997;
run;

proc sort data=INTERWRK.seinunit_naics_fnl2002_20;
  by sein year quarter es_naics_fnl2002;
run;

proc sort data=INTERWRK.seinunit_county_20;
  by sein year quarter es_county;
run;

proc sort data=INTERWRK.seinunit_owner_code_20;
  by sein year quarter es_owner_code;
run;

proc sort data=INTERWRK.seinunit_ein_20;
  by sein year quarter es_ein;
run;

proc sort data=INTERWRK.seinunit_wib_18;
  by sein year quarter leg_wib;
run;

proc sort data=INTERWRK.seinunit_msapmsa_18;
  by sein year quarter leg_msapmsa;
run;

proc sort data=INTERWRK.seinunit_state_18;
  by sein year quarter leg_state;
run;

proc sort data=INTERWRK.seinunit_county_18;
  by sein year quarter leg_county;
run;

proc sort data=INTERWRK.seinunit_subctygeo_18;
  by sein year quarter leg_subctygeo;
run;

```

Call the mode creation program for each variable .

```

%modevar(INTERWRK.seinunit_sic_20,step1,es_sic,4,%str('9999'));
%modevar(INTERWRK.seinunit_naics_eso1997_20,step2,es_naics_eso1997,6,%str('99999
9'));
%modevar(INTERWRK.seinunit_naics_eso2002_20,step3,es_naics_eso2002,6,%str('99999
9'));
%modevar(INTERWRK.seinunit_naics_fnl1997_20,step4,es_naics_fnl1997,6,%str('99999
9'));
%modevar(INTERWRK.seinunit_naics_fnl2002_20,step5,es_naics_fnl2002,6,%str('99999
9'));
%modevar(INTERWRK.seinunit_county_20,step6,es_county,3,%str('ZZZ'));
%modevar(INTERWRK.seinunit_owner_code_20,step7,es_owner_code,1,%str('9'));
%modevar(INTERWRK.seinunit_ein_20,step8,es_ein,9,%str(' '));
%modevar(INTERWRK.seinunit_wib_18,step9,leg_wib,6,%str(' '));
%modevar(INTERWRK.seinunit_msapmsa_18,step10,leg_msapmsa,8,%str(' '));
%modevar(INTERWRK.seinunit_state_18,step11,leg_state,2,%str(' '));
%modevar(INTERWRK.seinunit_county_18,step12,leg_county,3,%str(' '));
%modevar(INTERWRK.seinunit_subctygeo_18,step13,leg_subctygeo,10,%str(' '));

```

Merge everything back together.

```

data INTERWRK.sein_modes_21();
  merge step1 step2 step3 step4 step5 step6 step7 step8
        step9 step10 step11 step12 step13;
  by sein year quarter;
run;

```

Diagnostics moved to `ecfdiag_21.sas`

1.5.2.22 library/sasprogs/22_ecf.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/22_ecf.sas >--+ */
/*----- LEGACY NAME: 22_sein_wide.sas -----*/

```

This program transforms the modes calculated in program 21 into wide format.

Filling any remaining missing values requires a transformation of the data into wide format (one record for each SEIN with the YEAR QUARTER values for a variable stored in arrays). The wide file is then merged by SEIN with the data in the standard “long” format. The complete history for each SEIN SEINUNIT is then available to fill any missing values.

The following variables are transformed into wide format:

- mode_es_sic
- mode_es_naics
- mode_es_county
- mode_es_owner_code
- mode_es_ein
- mode_leg_wib
- mode_leg_msapmsa
- mode_leg_state
- mode_leg_county

- mode_leg_subctygeo.

There are actually two variables for each variable in the previous list, one for the unit weighted mode and an additional one for the employment weighted mode.

```

/* See runtime_ecf.sas for configuration options */
/* ALL macros are stored in runtime_ecf.sas */

/*****
/* REQUIRES-SAS: runtime_ecf.sas
/* REQUIRES-DATA: sein_modes_21.sas7bdat
/* PROVIDES-DATA: sein_wide_22.sas7bdat, sein_long_22.sas7bdat
*****/

* %include "runtime_ecf.sas";

title1 "Transform data from narrow to wide format";
title2 "Program Location: &curdir./22_sein_wide.sas";

```

Create the wide dataset .

```

data INTERWRK.sein_wide_22(keep=sein
                        %bldstr(sic,no)
                        %bldstr(sic_e,no)
                        %bldstr(neso1997,no)
                        %bldstr(neso1997_e,no)
                        %bldstr(neso2002,no)
                        %bldstr(neso2002_e,no)
                        %bldstr(naics1997,no)
                        %bldstr(naics1997_e,no)
                        %bldstr(naics2002,no)
                        %bldstr(naics2002_e,no)
                        %bldstr(fips,no)
                        %bldstr(fips_e,no)
                        %bldstr(owner_code,no)
                        %bldstr(owner_code_e,no)
                        %bldstr(en,yes)
                        %bldstr(en_e,yes)
                        %bldstr(leg_wib,no)
                        %bldstr(leg_wib_e,no)
                        %bldstr(leg_msapmsa,no)
                        %bldstr(leg_msapmsa_e,no)
                        %bldstr(leg_state,no)
                        %bldstr(leg_state_e,no)
                        %bldstr(leg_county,no)
                        %bldstr(leg_county_e,no)
                        %bldstr(leg_subctygeo,no)
                        %bldstr(leg_subctygeo_e,no))

INTERWRK.sein_long_22(keep=sein year quarter);
set INTERWRK.sein_modes_21();
  by sein year quarter;

```

Set the default length of each variable in the array .

```

/* SIC */
length %bldstr(sic,no) $4;
length %bldstr(sic_e,no) $4;

/* NAICS */
length %bldstr(neso1997,no) $6;
length %bldstr(neso1997_e,no) $6;
length %bldstr(neso2002,no) $6;
length %bldstr(neso2002_e,no) $6;

length %bldstr(naics1997,no) $6;
length %bldstr(naics1997_e,no) $6;
length %bldstr(naics2002,no) $6;
length %bldstr(naics2002_e,no) $6;

/* ES COUNTY */
length %bldstr(fips,no) $3;
length %bldstr(fips_e,no) $3;

/* OWNER_CODE */
length %bldstr(owner_code,no) $1;
length %bldstr(owner_code_e,no) $1;

/* EIN */
length %bldstr(en,yes) $9;
length %bldstr(en_e,yes) $9;

/* WIB */
length %bldstr(leg_wib,no) $6;
length %bldstr(leg_wib_e,no) $6;

/* MSAPMSA */
length %bldstr(leg_msapmsa,no) $8;
length %bldstr(leg_msapmsa_e,no) $8;

/* LEG STATE */
length %bldstr(leg_state,no) $2;
length %bldstr(leg_state_e,no) $2;

/* LEG COUNTY */
length %bldstr(leg_county,no) $3;
length %bldstr(leg_county_e,no) $3;

/* LEG SUBCTYGeo */
length %bldstr(leg_subctygeo,no) $10;
length %bldstr(leg_subctygeo_e,no) $10;

```

Retain the Variables in the array .

```
retain
  %bldstr(sic,no)
  %bldstr(sic_e,no)
  %bldstr(neso1997,no)
  %bldstr(neso1997_e,no)
  %bldstr(neso2002,no)
  %bldstr(neso2002_e,no)
  %bldstr(naics1997,no)
  %bldstr(naics1997_e,no)
  %bldstr(naics2002,no)
  %bldstr(naics2002_e,no)
  %bldstr(fips,no)
  %bldstr(fips_e,no)
  %bldstr(owner_code,no)
  %bldstr(owner_code_e,no)
  %bldstr(en,yes)
  %bldstr(en_e,yes)
  %bldstr(leg_wib,no)
  %bldstr(leg_wib_e,no)
  %bldstr(leg_msapmsa,no)
  %bldstr(leg_msapmsa_e,no)
  %bldstr(leg_state,no)
  %bldstr(leg_state_e,no)
  %bldstr(leg_county,no)
  %bldstr(leg_county_e,no)
  %bldstr(leg_subctygeo,no)
  %bldstr(leg_subctygeo_e,no)
;
```

Set up the arrays .

```

array sic{&start_year:&end_year,1:4} %bldstr(sic,no);
array sic_e{&start_year:&end_year,1:4} %bldstr(sic_e,no);

array nce97{&start_year:&end_year,1:4} %bldstr(neso1997,no);
array nce97_e{&start_year:&end_year,1:4} %bldstr(neso1997_e,no);

array nce02{&start_year:&end_year,1:4} %bldstr(neso2002,no);
array nce02_e{&start_year:&end_year,1:4} %bldstr(neso2002_e,no);

array ncf97{&start_year:&end_year,1:4} %bldstr(naics1997,no);
array ncf97_e{&start_year:&end_year,1:4} %bldstr(naics1997_e,no);

array ncf02{&start_year:&end_year,1:4} %bldstr(naics2002,no);
array ncf02_e{&start_year:&end_year,1:4} %bldstr(naics2002_e,no);

array fip{&start_year:&end_year,1:4} %bldstr(fips,no);
array fip_e{&start_year:&end_year,1:4} %bldstr(fips_e,no);

array own{&start_year:&end_year,1:4} %bldstr(owner_code,no);
array own_e{&start_year:&end_year,1:4} %bldstr(owner_code_e,no);

array en{&start_year:&end_year,1:4} %bldstr(en,yes);
array en_e{&start_year:&end_year,1:4} %bldstr(en_e,yes);

array wib{&start_year:&end_year,1:4} %bldstr(leg_wib,no);
array wib_e{&start_year:&end_year,1:4} %bldstr(leg_wib_e,no);

array msa{&start_year:&end_year,1:4} %bldstr(leg_msapmsa,no);
array msa_e{&start_year:&end_year,1:4} %bldstr(leg_msapmsa_e,no);

array st{&start_year:&end_year,1:4} %bldstr(leg_state,no);
array st_e{&start_year:&end_year,1:4} %bldstr(leg_state_e,no);

array cty{&start_year:&end_year,1:4} %bldstr(leg_county,no);
array cty_e{&start_year:&end_year,1:4} %bldstr(leg_county_e,no);

array sub{&start_year:&end_year,1:4} %bldstr(leg_subctygeo,no);
array sub_e{&start_year:&end_year,1:4} %bldstr(leg_subctygeo_e,no);

```

Initialize the arrays .

```

if first.sein then do;
  do i=&start_year to &end_year;
    do j=1 to 4;
      sic{i,j}="9999";
      sic_e{i,j}="9999";
      nce97{i,j}="999999";
      nce97_e{i,j}="999999";
      nce02{i,j}="999999";
      nce02_e{i,j}="999999";
      ncf97{i,j}="999999";
      ncf97_e{i,j}="999999";
      ncf02{i,j}="999999";
      ncf02_e{i,j}="999999";
      fip{i,j}="ZZZ";
      fip_e{i,j}="ZZZ";
      own{i,j}="9";
      own_e{i,j}="9";
      en{i,j}=" ";
      en_e{i,j}=" ";
      wib{i,j}=" ";
      wib_e{i,j}=" ";
      msa{i,j}=" ";
      msa_e{i,j}=" ";
      st{i,j}=" ";
      st_e{i,j}=" ";
      cty{i,j}=" ";
      cty_e{i,j}=" ";
      sub{i,j}=" ";
      sub_e{i,j}=" ";
    end;
  end;
end;

```

Load the arrays .

```

    sic{year,quarter}=mode_es_sic;
    sic_e{year,quarter}=mode_es_sic_emp;
    nce97{year,quarter}=mode_es_naics_eso1997;
    nce97_e{year,quarter}=mode_es_naics_eso1997_emp;
    nce02{year,quarter}=mode_es_naics_eso2002;
    nce02_e{year,quarter}=mode_es_naics_eso2002_emp;
    ncf97{year,quarter}=mode_es_naics_fnl1997;
    ncf97_e{year,quarter}=mode_es_naics_fnl1997_emp;
    ncf02{year,quarter}=mode_es_naics_fnl2002;
    ncf02_e{year,quarter}=mode_es_naics_fnl2002_emp;
    fip{year,quarter}=mode_es_county;
    fip_e{year,quarter}=mode_es_county_emp;
    own{year,quarter}=mode_es_owner_code;
    own_e{year,quarter}=mode_es_owner_code_emp;
    en{year,quarter}=mode_es_ein;
    en_e{year,quarter}=mode_es_ein_emp;
    wib{year,quarter}=mode_leg_wib;
    wib_e{year,quarter}=mode_leg_wib_emp;
    msa{year,quarter}=mode_leg_msapmsa;
    msa_e{year,quarter}=mode_leg_msapmsa_emp;
    st{year,quarter}=mode_leg_state;
    st_e{year,quarter}=mode_leg_state_emp;
    cty{year,quarter}=mode_leg_county;
    cty_e{year,quarter}=mode_leg_county_emp;
    sub{year,quarter}=mode_leg_subctygeo;
    sub_e{year,quarter}=mode_leg_subctygeo_emp;

    if last.sein then output INTERWRK.sein_wide_22;
    output INTERWRK.sein_long_22;
run;

```

Sort the data.

```

proc sort data=INTERWRK.sein_wide_22;
    by sein;
run;

```

Diagnostics moved to ecfdiag_22.sas

1.5.2.23 library/sasprogs/23_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/sasprogs/23_ecf.sas >--- */
/*----- LEGACY NAME: 23_sein_fill.sas -----*/

```

This program fills missing values in the SEIN level mode variables.

The variables in program 22 that were transformed into wide format are filled, if missing, with the closest value from another YEAR QUARTER.

The same logic is used in this program as in program 19. Missing values need to be filled for the GEO variables at the SEIN level even though they contain no missing values at the SEINUNIT level. For quarters where no SEINUNITS are active, the employment weighted mode will be missing. This value is filled, if available, with information from another quarter.

```

/*****
/* REQUIRES-DATA: sein_long_22.sas7bdat, sein_wide_22.sas7bdat */
/* PROVIDES-DATA: sein_missing_23.sas7bdat, sein_mode_fill_23.sas7bdat */
*****/

title1 "Fill at the SEIN level";
title2 "Program Location: &curdir./23_fill_sync.sas";

```

Begin the fill.

```
data INTERWRK.sein_mode_fill_23
  (keep=sein year quarter _merge
    mode_es_sic mode_es_sic_miss mode_es_sic_flag
    mode_es_sic_emp mode_es_sic_emp_miss mode_es_sic_emp_flag
    mode_es_naics_eso1997 mode_es_naics_eso1997_miss
mode_es_naics_eso1997_flag
    mode_es_naics_eso1997_emp mode_es_naics_eso1997_emp_miss
mode_es_naics_eso1997_emp_flag
    mode_es_naics_eso2002 mode_es_naics_eso2002_miss
mode_es_naics_eso2002_flag
    mode_es_naics_eso2002_emp mode_es_naics_eso2002_emp_miss
mode_es_naics_eso2002_emp_flag
    mode_es_naics_fnl1997 mode_es_naics_fnl1997_miss
mode_es_naics_fnl1997_flag
    mode_es_naics_fnl1997_emp mode_es_naics_fnl1997_emp_miss
mode_es_naics_fnl1997_emp_flag
    mode_es_naics_fnl2002 mode_es_naics_fnl2002_miss
mode_es_naics_fnl2002_flag
    mode_es_naics_fnl2002_emp mode_es_naics_fnl2002_emp_miss
mode_es_naics_fnl2002_emp_flag
    mode_es_county mode_es_county_miss mode_es_county_flag
    mode_es_county_emp mode_es_county_emp_miss mode_es_county_emp_flag
mode_es_owner_code mode_es_owner_code_miss mode_es_owner_code_flag
    mode_es_owner_code_emp mode_es_owner_code_emp_miss
mode_es_owner_code_emp_flag
    mode_es_ein mode_es_ein_miss mode_es_ein_flag
    mode_es_ein_emp mode_es_ein_emp_miss mode_es_ein_emp_flag
    mode_leg_wib mode_leg_wib_miss mode_leg_wib_flag
    mode_leg_wib_emp mode_leg_wib_emp_miss mode_leg_wib_emp_flag
    mode_leg_msapmsa mode_leg_msapmsa_miss mode_leg_msapmsa_flag
    mode_leg_msapmsa_emp mode_leg_msapmsa_emp_miss
mode_leg_msapmsa_emp_flag
    mode_leg_state mode_leg_state_miss mode_leg_state_flag
    mode_leg_state_emp mode_leg_state_emp_miss mode_leg_state_emp_flag
    mode_leg_county mode_leg_county_miss mode_leg_county_flag
    mode_leg_county_emp mode_leg_county_emp_miss mode_leg_county_emp_flag
    mode_leg_subctygeo mode_leg_subctygeo_miss mode_leg_subctygeo_flag
    mode_leg_subctygeo_emp mode_leg_subctygeo_emp_miss
mode_leg_subctygeo_emp_flag
  )
  INTERWRK.sein_missing_23(keep=sein);

merge INTERWRK.sein_long_22(in=_mvar1) INTERWRK.sein_wide_22(in=_mvar2);
  by sein;
```

Length statements.

```

length MODE_ES_SIC $4
      MODE_ES_COUNTY $3
      MODE_ES_EIN $9
      MODE_ES_NAICS_ESO1997 MODE_ES_NAICS_ESO2002 MODE_ES_NAICS_FNL1997
      MODE_ES_NAICS_FNL2002 $6
      MODE_ES_OWNER_CODE $1
      MODE_ES_SIC_EMP $4
      MODE_ES_COUNTY_EMP $3
      MODE_ES_EIN_EMP $9
      MODE_ES_NAICS_ESO1997_EMP MODE_ES_NAICS_ESO2002_EMP
      MODE_ES_NAICS_FNL1997_EMP MODE_ES_NAICS_FNL2002_EMP $6
      MODE_ES_OWNER_CODE_EMP $1
      MODE_LEG_WIB MODE_LEG_WIB_EMP $6
      MODE_LEG_MSAPMSA MODE_LEG_MSAPMSA_EMP $8
      MODE_LEG_STATE MODE_LEG_STATE_EMP $2
      MODE_LEG_COUNTY MODE_LEG_COUNTY_EMP $3
      MODE_LEG_SUBCTYGEO MODE_LEG_SUBCTYGEO_EMP $10
;

```

Merge control a la Stata.

```

if _mvar1=1 and _mvar2=1 then _merge=3;
if _mvar1=1 and _mvar2=0 then _merge=1;
if _mvar1=0 and _mvar2=1 then _merge=2;

```

Define and load the arrays.

```

array sc{&start_year:&end_year,1:4} %bldstr(sic,no);
array sc_e{&start_year:&end_year,1:4} %bldstr(sic_e,no);

array nce97{&start_year:&end_year,1:4} %bldstr(neso1997,no);
array nce97_e{&start_year:&end_year,1:4} %bldstr(neso1997_e,no);

array nce02{&start_year:&end_year,1:4} %bldstr(neso2002,no);
array nce02_e{&start_year:&end_year,1:4} %bldstr(neso2002_e,no);

array ncf97{&start_year:&end_year,1:4} %bldstr(naics1997,no);
array ncf97_e{&start_year:&end_year,1:4} %bldstr(naics1997_e,no);

array ncf02{&start_year:&end_year,1:4} %bldstr(naics2002,no);
array ncf02_e{&start_year:&end_year,1:4} %bldstr(naics2002_e,no);

array fip{&start_year:&end_year,1:4} %bldstr(fips,no);
array fip_e{&start_year:&end_year,1:4} %bldstr(fips_e,no);

array own{&start_year:&end_year,1:4} %bldstr(owner_code,no);
array own_e{&start_year:&end_year,1:4} %bldstr(owner_code_e,no);

array en{&start_year:&end_year,1:4} %bldstr(en,yes);
array en_e{&start_year:&end_year,1:4} %bldstr(en_e,yes);

array wib{&start_year:&end_year,1:4} %bldstr(leg_wib,no);
array wib_e{&start_year:&end_year,1:4} %bldstr(leg_wib_e,no);

array msa{&start_year:&end_year,1:4} %bldstr(leg_msapmsa,no);
array msa_e{&start_year:&end_year,1:4} %bldstr(leg_msapmsa_e,no);

array st{&start_year:&end_year,1:4} %bldstr(leg_state,no);
array st_e{&start_year:&end_year,1:4} %bldstr(leg_state_e,no);

array cty{&start_year:&end_year,1:4} %bldstr(leg_county,no);
array cty_e{&start_year:&end_year,1:4} %bldstr(leg_county_e,no);

array sub{&start_year:&end_year,1:4} %bldstr(leg_subctygeo,no);
array sub_e{&start_year:&end_year,1:4} %bldstr(leg_subctygeo_e,no);

```

Assign the data to the appropriate year .

```
MODE_ES_SIC=sc{year,quarter};
MODE_ES_SIC_EMP=sc_e{year,quarter};
MODE_ES_NAICS_ES01997=nce97{year,quarter};
MODE_ES_NAICS_ES01997_EMP=nce97_e{year,quarter};
MODE_ES_NAICS_ES02002=nce02{year,quarter};
MODE_ES_NAICS_ES02002_EMP=nce02_e{year,quarter};
MODE_ES_NAICS_FNL1997=ncf97{year,quarter};
MODE_ES_NAICS_FNL1997_EMP=ncf97_e{year,quarter};
MODE_ES_NAICS_FNL2002=ncf02{year,quarter};
MODE_ES_NAICS_FNL2002_EMP=ncf02_e{year,quarter};
MODE_ES_COUNTY=fip{year,quarter};
MODE_ES_COUNTY_EMP=fip_e{year,quarter};
MODE_ES_OWNER_CODE=own{year,quarter};
MODE_ES_OWNER_CODE_EMP=own_e{year,quarter};
MODE_ES_EIN=en{year,quarter};
MODE_ES_EIN_EMP=en_e{year,quarter};
MODE_LEG_WIB=wib{year,quarter};
MODE_LEG_WIB_EMP=wib_e{year,quarter};
MODE_LEG_MSAPMSA=msa{year,quarter};
MODE_LEG_MSAPMSA_EMP=msa_e{year,quarter};
MODE_LEG_STATE=st{year,quarter};
MODE_LEG_STATE_EMP=st_e{year,quarter};
MODE_LEG_COUNTY=cty{year,quarter};
MODE_LEG_COUNTY_EMP=cty_e{year,quarter};
MODE_LEG_SUBCTYGEO=sub{year,quarter};
MODE_LEG_SUBCTYGEO_EMP=sub_e{year,quarter};
```

Process Unit Weighted Variables

Set flag if data is missing.

```

mode_es_sic_miss=0;
mode_es_naics_eso1997_miss=0;
mode_es_naics_eso2002_miss=0;
mode_es_naics_fnl1997_miss=0;
mode_es_naics_fnl2002_miss=0;
mode_es_county_miss=0;
mode_es_owner_code_miss=0;
mode_es_ein_miss=0;
mode_leg_wib_miss=0;
mode_leg_msapmsa_miss=0;
mode_leg_state_miss=0;
mode_leg_county_miss=0;
mode_leg_subctygeo_miss=0;
if mode_es_sic='9999' then do;
    mode_es_sic_miss=1;
end;
if mode_es_naics_eso1997='999999' then do;
    mode_es_naics_eso1997_miss=1;
end;
if mode_es_naics_eso2002='999999' then do;
    mode_es_naics_eso2002_miss=1;
end;
if mode_es_naics_fnl1997='999999' then do;
    mode_es_naics_fnl1997_miss=1;
end;
if mode_es_naics_fnl2002='999999' then do;
    mode_es_naics_fnl2002_miss=1;
end;
if mode_es_county='ZZZ' then do;
    mode_es_county_miss=1;
end;
if mode_es_owner_code='9' then do;
    mode_es_owner_code_miss=1;
end;
if mode_es_ein=" " then do;
    mode_es_ein_miss=1;
end;
if mode_leg_wib=" " then do;
    mode_leg_wib_miss=1;
end;
if mode_leg_msapmsa=" " then do;
    mode_leg_msapmsa_miss=1;
end;
if mode_leg_state=" " then do;
    mode_leg_state_miss=1;
end;
if mode_leg_county=" " then do;
    mode_leg_county_miss=1;
end;
if mode_leg_subctygeo=" " then do;
    mode_leg_subctygeo_miss=1;
end;

```

ES_SIC cleanup.

```

if mode_es_sic_miss=1 then do;
    %qloop(mode_es_sic,sc,%str('9999'));
end;
else do;
    mode_es_sic_flag=0;
end;
if mode_es_sic_miss=1 and mode_es_sic~="9999" then mode_es_sic_miss=2;

```

ES_NAICS_ESO1997 cleanup.

```

if mode_es_naics_eso1997_miss=1 then do;
    %qloop(mode_es_naics_eso1997,nce97,%str('999999'));
end;
else do;
    mode_es_naics_eso1997_flag=0;
end;
if mode_es_naics_eso1997_miss=1 and mode_es_naics_eso1997~="999999" then
mode_es_naics_eso1997_miss=2;

```

ES_NAICS_ESO2002 cleanup.

```

if mode_es_naics_eso2002_miss=1 then do;
    %qloop(mode_es_naics_eso2002,nce02,%str('999999'));
end;
else do;
    mode_es_naics_eso2002_flag=0;
end;
if mode_es_naics_eso2002_miss=1 and mode_es_naics_eso2002~="999999" then
mode_es_naics_eso2002_miss=2;

```

ES_NAICS_FNL1997 cleanup .

```

if mode_es_naics_fnl1997_miss=1 then do;
    %qloop(mode_es_naics_fnl1997,ncf97,%str('999999'));
end;
else do;
    mode_es_naics_fnl1997_flag=0;
end;
if mode_es_naics_fnl1997_miss=1 and mode_es_naics_fnl1997~="999999" then
mode_es_naics_fnl1997_miss=2;

```

ES_NAICS_FNL2002 cleanup .

```

if mode_es_naics_fnl2002_miss=1 then do;
    %qloop(mode_es_naics_fnl2002,ncf02,%str('999999'));
end;
else do;
    mode_es_naics_fnl2002_flag=0;
end;
if mode_es_naics_fnl2002_miss=1 and mode_es_naics_fnl2002~="999999" then
mode_es_naics_fnl2002_miss=2;

```

ES_COUNTY cleanup.

```

if mode_es_county_miss=1 then do;
    %qloop(mode_es_county,fip,%str('ZZZ'));
end;
else do;
    mode_es_county_flag=0;
end;
if mode_es_county_miss=1 and mode_es_county~="ZZZ" then
mode_es_county_miss=2;

```

ES_OWNER_CODE cleanup .

```

if mode_es_owner_code_miss=1 then do;
    %qloop(mode_es_owner_code,own,%str('9'));
end;
else do;
    mode_es_owner_code_flag=0;
end;
if mode_es_owner_code_miss=1 and mode_es_owner_code~="9" then
mode_es_owner_code_miss=2;

```

EIN cleanup .

```
if mode_es_ein_miss=1 then do;
%macro ckkein;
  %if &einavail=ein %then %do;
    %qloop(mode_es_ein,en,%str(' '));
  %end;
%mend;
%ckkein;
end;
else do;
  mode_es_ein_flag=0;
end;
if mode_es_ein_miss=1 and mode_es_ein~=" " then mode_es_ein_miss=2;
```

WIB cleanup.

```
if mode_leg_wib_miss=1 then do;
  %qloop(mode_leg_wib,wib,%str(' '));
end;
else do;
  mode_leg_wib_flag=0;
end;
if mode_leg_wib_miss=1 and mode_leg_wib~=" " then mode_leg_wib_miss=2;
```

MSAPMSA cleanup.

```
if mode_leg_msapmsa_miss=1 then do;
  %qloop(mode_leg_msapmsa,msa,%str(' '));
end;
else do;
  mode_leg_msapmsa_flag=0;
end;
if mode_leg_msapmsa_miss=1 and mode_leg_msapmsa~=" " then
mode_leg_msapmsa_miss=2;
```

LEG STATE cleanup.

```
if mode_leg_state_miss=1 then do;
  %qloop(mode_leg_state,st,%str(' '));
end;
else do;
  mode_leg_state_flag=0;
end;
if mode_leg_state_miss=1 and mode_leg_state~=" " then
mode_leg_state_miss=2;
```

LEG COUNTY cleanup.

```
if mode_leg_county_miss=1 then do;
  %qloop(mode_leg_county,cty,%str(' '));
end;
else do;
  mode_leg_county_flag=0;
end;
if mode_leg_county_miss=1 and mode_leg_county~=" " then
mode_leg_county_miss=2;
```

SUBCTYGEO cleanup.

```
if mode_leg_subctygeo_miss=1 then do;
  %qloop(mode_leg_subctygeo,sub,%str(' '));
end;
else do;
  mode_leg_subctygeo_flag=0;
end;
if mode_leg_subctygeo_miss=1 and mode_leg_subctygeo~=" " then
mode_leg_subctygeo_miss=2;
```

Process Employment Weighted Variables.

Set flag if data is missing .

```
mode_es_sic_emp_miss=0;
mode_es_naics_eso1997_emp_miss=0;
mode_es_naics_eso2002_emp_miss=0;
mode_es_naics_fnl1997_emp_miss=0;
mode_es_naics_fnl2002_emp_miss=0;
mode_es_county_emp_miss=0;
mode_es_owner_code_emp_miss=0;
mode_es_ein_emp_miss=0;
mode_leg_wib_emp_miss=0;
mode_leg_msapmsa_emp_miss=0;
mode_leg_state_emp_miss=0;
mode_leg_county_emp_miss=0;
mode_leg_subctygeo_emp_miss=0;
if mode_es_sic_emp='9999' then do;
    mode_es_sic_emp_miss=1;
end;
if mode_es_naics_eso1997_emp='999999' then do;
    mode_es_naics_eso1997_emp_miss=1;
end;
if mode_es_naics_eso2002_emp='999999' then do;
    mode_es_naics_eso2002_emp_miss=1;
end;
if mode_es_naics_fnl1997_emp='999999' then do;
    mode_es_naics_fnl1997_emp_miss=1;
end;
if mode_es_naics_fnl2002_emp='999999' then do;
    mode_es_naics_fnl2002_emp_miss=1;
end;
if mode_es_county_emp='ZZZ' then do;
    mode_es_county_emp_miss=1;
end;
if mode_es_owner_code_emp='9' then do;
    mode_es_owner_code_emp_miss=1;
end;
if mode_es_ein_emp=" " then do;
    mode_es_ein_emp_miss=1;
end;
if mode_leg_wib_emp=" " then do;
    mode_leg_wib_emp_miss=1;
end;
if mode_leg_msapmsa_emp=" " then do;
    mode_leg_msapmsa_emp_miss=1;
end;
if mode_leg_state_emp=" " then do;
    mode_leg_state_emp_miss=1;
end;
if mode_leg_county_emp=" " then do;
    mode_leg_county_emp_miss=1;
end;
if mode_leg_subctygeo_emp=" " then do;
    mode_leg_subctygeo_emp_miss=1;
end;
```

ES_SIC cleanup.

```

if mode_es_sic_emp_miss=1 then do;
    %qloop(mode_es_sic_emp,sc_e,%str('9999'));
end;
else do;
    mode_es_sic_emp_flag=0;
end;
if mode_es_sic_emp_miss=1 and mode_es_sic_emp~="9999" then
mode_es_sic_emp_miss=2;

```

ES_NAICS_ESO1997 cleanup.

```

if mode_es_naics_eso1997_emp_miss=1 then do;
    %qloop(mode_es_naics_eso1997_emp,nce97_e,%str('999999'));
end;
else do;
    mode_es_naics_eso1997_emp_flag=0;
end;
if mode_es_naics_eso1997_emp_miss=1 and
mode_es_naics_eso1997_emp~="999999"
then mode_es_naics_eso1997_emp_miss=2;

```

ES_NAICS_ESO2002 cleanup.

```

if mode_es_naics_eso2002_emp_miss=1 then do;
    %qloop(mode_es_naics_eso2002_emp,nce02_e,%str('999999'));
end;
else do;
    mode_es_naics_eso2002_emp_flag=0;
end;
if mode_es_naics_eso2002_emp_miss=1 and
mode_es_naics_eso2002_emp~="999999"
then mode_es_naics_eso2002_emp_miss=2;

```

ES_NAICS_FNL1997 cleanup.

```

if mode_es_naics_fnl1997_emp_miss=1 then do;
    %qloop(mode_es_naics_fnl1997_emp,ncf97_e,%str('999999'));
end;
else do;
    mode_es_naics_fnl1997_emp_flag=0;
end;
if mode_es_naics_fnl1997_emp_miss=1 and
mode_es_naics_fnl1997_emp~="999999"
then mode_es_naics_fnl1997_emp_miss=2;

```

ES_NAICS_FNL2002 cleanup.

```

if mode_es_naics_fnl2002_emp_miss=1 then do;
    %qloop(mode_es_naics_fnl2002_emp,ncf02_e,%str('999999'));
end;
else do;
    mode_es_naics_fnl2002_emp_flag=0;
end;
if mode_es_naics_fnl2002_emp_miss=1 and
mode_es_naics_fnl2002_emp~="999999"
then mode_es_naics_fnl2002_emp_miss=2;

```

ES_COUNTY cleanup.

```

if mode_es_county_emp_miss=1 then do;
    %qloop(mode_es_county_emp,fip_e,%str('ZZZ'));
end;
else do;
    mode_es_county_emp_flag=0;
end;
if mode_es_county_emp_miss=1 and mode_es_county_emp~="ZZZ"
then mode_es_county_emp_miss=2;

```

ES_OWNER_CODE cleanup.

```
if mode_es_owner_code_emp_miss=1 then do;
    %qloop(mode_es_owner_code_emp,own_e,%str('9'));
end;
else do;
    mode_es_owner_code_emp_flag=0;
end;
if mode_es_owner_code_emp_miss=1 and mode_es_owner_code_emp~="9"
    then mode_es_owner_code_emp_miss=2;
```

EIN cleanup.

```
if mode_es_ein_emp_miss=1 then do;
%macro ckkein;
    %if &einavail=ein %then %do;
        %qloop(mode_es_ein_emp,en_e,%str(' '));
    %end;
%mend;
%ckkein;
end;
else do;
    mode_es_ein_emp_flag=0;
end;
if mode_es_ein_emp_miss=1 and mode_es_ein_emp~=" " then
mode_es_ein_emp_miss=2;
```

WIB cleanup.

```
if mode_leg_wib_emp_miss=1 then do;
    %qloop(mode_leg_wib_emp,wib_e,%str(' '));
end;
else do;
    mode_leg_wib_emp_flag=0;
end;
if mode_leg_wib_emp_miss=1 and mode_leg_wib_emp~=" " then
mode_leg_wib_emp_miss=2;
```

MSAPMSA cleanup.

```
if mode_leg_msapmsa_emp_miss=1 then do;
    %qloop(mode_leg_msapmsa_emp,msa_e,%str(' '));
end;
else do;
    mode_leg_msapmsa_emp_flag=0;
end;
if mode_leg_msapmsa_emp_miss=1 and mode_leg_msapmsa_emp~=" "
    then mode_leg_msapmsa_emp_miss=2;
```

LEG STATE cleanup.

```
if mode_leg_state_emp_miss=1 then do;
    %qloop(mode_leg_state_emp,st_e,%str(' '));
end;
else do;
    mode_leg_state_emp_flag=0;
end;
if mode_leg_state_emp_miss=1 and mode_leg_state_emp~=" "
    then mode_leg_state_emp_miss=2;
```

LEG COUNTY cleanup.

```

if mode_leg_county_emp_miss=1 then do;
    %qloop(mode_leg_county_emp,cty_e,%str(' '));
end;
else do;
    mode_leg_county_emp_flag=0;
end;
if mode_leg_county_emp_miss=1 and mode_leg_county_emp~=" "
    then mode_leg_county_emp_miss=2;

```

SUBCTYGEO cleanup.

```

if mode_leg_subctygeo_emp_miss=1 then do;
    %qloop(mode_leg_subctygeo_emp,sub_e,%str(' '));
end;
else do;
    mode_leg_subctygeo_emp_flag=0;
end;
if mode_leg_subctygeo_emp_miss=1 and mode_leg_subctygeo_emp~=" " then
mode_leg_subctygeo_emp_miss=2;

if mode_es_sic_miss=1 or mode_es_sic_emp_miss=1 then output
INTERWRK.sein_missing_23;
output INTERWRK.sein_mode_fill_23;
run;

```

Diagnostics moved to ecfdiag_23.sas

```

proc sort data=INTERWRK.sein_mode_fill_23;
    by sein year quarter;
run;

proc sort data=INTERWRK.sein_missing_23 nodupkey;
    by sein;
run;

```

1.5.2.24 library/sasprogs/24_ecf.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/24_ecf.sas >--+ */
/*----- LEGACY NAME: 24_impute_sein_industry.sas -----*/

```

This program fills missing SEIN values with the unit weighted mode if available. SIC, if still missing is imputed.

Missing SEIN YEAR QUARTER values for SIC, NAICS, county, ownership code, and EIN are filled with the unit weighted mode.

SIC, must not have any missing values for QWI processing to proceed. Any SIC missing values that remain are for firms that never reported either a NAICS or SIC code over the entire time period. Since we have little information, the distribution of employment across 4 digit SIC is used. A random number is drawn that determines the actual imputed SIC. SIC codes with a larger share of employment are more at risk for assignment than those with a relatively small share. The SIC is imputed once for each SEIN and thus remains constant across time.

```

/*****
/* REQUIRES-DATA: sein_mode_fill_23.sas7bdat */
/* PROVIDES-DATA: sein_missing_24.sas7bdat */
/*      sein_long_24.sas7bdat */
/*      sicfreq_24.sas7bdat */
/*      sicfreq2_24.sas7bdat */
*****/

title1 "Do the Unconditional Impute";
title2 "Program Location: &curdir./24_impute_sein_industry.sas";

```

Get the Frequency of Each 4 digit SIC code .

```

%macro sicdate;
%let decstart=%eval(&start_year.*10 + &start_quarter);
%if ( &decstart > 19971 ) %then %do;
  %let sicimpyear=&start_year.;
  %let sicimpquarter=&start_quarter.;
%end;
%else %do;
  %let sicimpyear=1997;
  %let sicimpquarter=1;
%end;

proc freq data=INTERWRK.sein_mode_fill_23(where=(mode_es_sic_emp~="9999" and
year=&sicimpyear and quarter=&sicimpquarter));
  tables mode_es_sic_emp /out=INTERWRK.sicfreq_24;
run;
%mend sicdate;
%sicdate;

/* diagnostics moved to ecfdiag_24.sas */

```

Create the necessary begin and end range variables .

```

data INTERWRK.sicfreq2_24;
  set INTERWRK.sicfreq_24(keep=mode_es_sic_emp percent);

  begin_value=cum_percent;
  cum_percent+(percent*.01);
  end_value=cum_percent;
run;

proc print data=INTERWRK.sicfreq2_24;
run;

```

Output the format.

```

data temp1;
  set INTERWRK.sicfreq2_24 end=lastrec ;

  file "%sysfunc(pathname(interwrk))/24_impute_sic_format.sas";

  if _N_=1 then do;
    put "*** Format for SIC Impute ***";
    put " ";
    put "proc format;" ;
    put " value sicimp";
    put "      begin_value "<- " end_value "=" mode_es_sic_emp;
  end;
  else do;
    put "      begin_value "<- " end_value "=" mode_es_sic_emp;
  end;

  if lastrec=1 then do;
    put ";";
    put "run;";
  end;
run;

```

Impute the SIC and NAICS.

```

%include "%sysfunc(pathname(interwrk))/24_impute_sic_format.sas";

data INTERWRK.sein_missing_24;
  set INTERWRK.sein_missing_23;

  length sic_impute $4;

  uniform0=uniform(49237543);
  sic_impute=put(uniform0,sicimp.);

run;

/* diagnostics moved to ecfdiag_24.sas */

```

Merge the imputed SIC and NAICS back onto the SEIN mode values .

```

data INTERWRK.sein_long_24(drop=uniform0 uniform1 uniform2 sic_impute continue
  continuel EMP employ
  es_naics1997 es_sic found_ind_temp lookup_temp
  low_limit pct_emp sum_n97 sum_sic up_limit
  mode_es_naics_fnl1997_emp_clean naics1997_impute
  mode_es_sic_emp_temp mode_es_naics_fnl2002_emp_clean
  naics2002_impute mode_es_naics_fnl1997_emp_temp);
  merge INTERWRK.sein_mode_fill_23(in=a drop=_merge)
        INTERWRK.sein_missing_24(in=b);
  by sein;

  if a=1 and b=1 then _merge=3;
  if a=1 and b=0 then _merge=1;
  if a=0 and b=1 then _merge=2;

  retain uniform1 uniform2;

```

Fill first with unit weighted mode if available .

```

if mode_es_sic_emp="9999" and mode_es_sic~="9999" then do;
  mode_es_sic_emp=mode_es_sic;
  mode_es_sic_emp_miss=5;
end;
if mode_es_sic="9999" then do;
  mode_es_sic=sic_impute;
  mode_es_sic_miss=6;
end;
if mode_es_sic_emp="9999" then do;
  mode_es_sic_emp=sic_impute;
  mode_es_sic_emp_miss=6;
end;

```

Fill first with unit weighted mode if available .

```

if mode_es_naics_fnl1997_emp="999999" and mode_es_naics_fnl1997~="999999"
then do;
  mode_es_naics_fnl1997_emp=mode_es_naics_fnl1997;
  mode_es_naics_fnl1997_emp_miss=5;
end;
if mode_es_naics_fnl2002_emp="999999" and mode_es_naics_fnl2002~="999999"
then do;
  mode_es_naics_fnl2002_emp=mode_es_naics_fnl2002;
  mode_es_naics_fnl2002_emp_miss=5;
end;

```

Fill first with unit weighted mode if available .

```

if mode_es_naics_eso1997_emp="999999" and mode_es_naics_eso1997~="999999"
then do;
  mode_es_naics_eso1997_emp=mode_es_naics_eso1997;
  mode_es_naics_eso1997_emp_miss=5;
end;
if mode_es_naics_eso2002_emp="999999" and mode_es_naics_eso2002~="999999"
then do;
  mode_es_naics_eso2002_emp=mode_es_naics_eso2002;
  mode_es_naics_eso2002_emp_miss=5;
end;

```

Use SIC Impute Value to impute NAICS1997 and then impute NAICS2002 .

```

if first.sein then do;
  uniform1=uniform(8364676);
  uniform2=uniform(4453234);
end;

```

Fill NAICS1997 Missings using SIC

```

*****;
%indlkup2(mode_es_naics_fnl1997_emp,mode_es_sic_emp,6,4,
  es_sic,naics1997_impute,uniform1,parms_us_imp_ncs1997_sic);
if mode_es_naics_fnl1997_emp_miss=3 or mode_es_naics_fnl1997_emp_miss=4
then do;
  mode_es_naics_fnl1997_emp_miss=6;
  mode_es_naics_fnl1997_miss=6;
  mode_es_naics_fnl1997=mode_es_naics_fnl1997_emp;
end;

```

Fill NAICS2002 Missings using NAICS1997

```

%indlkup2(mode_es_naics_fnl2002_emp,mode_es_naics_fnl1997_emp,6,6,
es_naics1997,naics2002_impute,uniform2,parms_us_imp_ncs2002_ncs1997);
if mode_es_naics_fnl2002_emp_miss=3 or mode_es_naics_fnl2002_emp_miss=4
then do;
mode_es_naics_fnl2002_emp_miss=6;
mode_es_naics_fnl2002_miss=6;
mode_es_naics_fnl2002=mode_es_naics_fnl2002_emp;
end;

```

Fill NAICS1997 Missings using SIC.

```

%indlkup2(mode_es_naics_eso1997_emp,mode_es_sic_emp,6,4,
es_sic,naics1997_impute,uniform1,parms_us_imp_ncs1997_sic);
if mode_es_naics_eso1997_emp_miss=3 or mode_es_naics_eso1997_emp_miss=4
then do;
mode_es_naics_eso1997_emp_miss=6;
mode_es_naics_eso1997_miss=6;
mode_es_naics_eso1997=mode_es_naics_eso1997_emp;
end;

```

Fill NAICS2002 Missings using NAICS1997.

```

%indlkup2(mode_es_naics_eso2002_emp,mode_es_naics_eso1997_emp,6,6,
es_naics1997,naics2002_impute,uniform2,parms_us_imp_ncs2002_ncs1997);
if mode_es_naics_eso2002_emp_miss=3 or mode_es_naics_eso2002_emp_miss=4
then do;
mode_es_naics_eso2002_emp_miss=6;
mode_es_naics_eso2002_miss=6;
mode_es_naics_eso2002=mode_es_naics_eso2002_emp;
end;

```

Fill other mode values similarly, Except No impute.

```

if mode_es_county_emp="ZZZ" and mode_es_county~="ZZZ" then do;
mode_es_county_emp=mode_es_county;
mode_es_county_emp_miss=5;
end;
if mode_es_owner_code_emp="9" and mode_es_owner_code~="9" then do;
mode_es_owner_code_emp=mode_es_owner_code;
mode_es_owner_code_emp_miss=5;
end;
if mode_es_ein_emp=" " and mode_es_ein~=" " then do;
mode_es_ein_emp=mode_es_ein;
mode_es_ein_emp_miss=5;
end;
run;

```

diagnostics moved to ecfdiag_24.sas

1.5.2.25 library/sasprogs/25_ecf.sas

```

/* +---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* +---< Location: /programs/production/dev1/current/ecf >---+ */
/* +---< File: library/sasprogs/25_ecf.sas >---+ */
/*----- LEGACY NAME: 25_impute_seinunit_industry.sas -----*/

```

This is a simple program that fills the SEINUNIT missing values with the SEIN mode. Except for SIC and NAICS some missing values will still remain for firms that never reported any information for that variable.

```

/*****
/* REQUIRES-DATA: seinunit_long_20.sas7bdat, sein_long_24.sas7bdat */
/* PROVIDES-DATA: seinunit_long_25.sas7bdat */
*****/

title1 "Fill the SEINUNIT data";
title2 "Program Location: &curdir./25_impute_seinunit_industry.sas";

```

Bring the SEINUNIT and SEIN Long Files together .

```

data INTERWRK.seinunit_long_25;
  merge INTERWRK.seinunit_long_20(in=a drop=merge) INTERWRK.sein_long_24(in=b
drop=_merge);
  by sein year quarter;

```

Stata-like merge control.

```

if a=1 and b=1 then merge=3;
if a=1 and b=0 then merge=1;
if a=0 and b=1 then merge=2;

```

SIC.

```

if es_sic_miss=1 then do;
  es_sic=mode_es_sic_emp;
  es_sic_miss=mode_es_sic_emp_miss+5;
end;

```

NAICS.

```

if es_naics_fnl1997_miss=1 then do;
  es_naics_fnl1997=mode_es_naics_fnl1997_emp;
  es_naics_fnl1997_miss=mode_es_naics_fnl1997_emp_miss+5;
end;
if es_naics_fnl2002_miss=1 then do;
  es_naics_fnl2002=mode_es_naics_fnl2002_emp;
  es_naics_fnl2002_miss=mode_es_naics_fnl2002_emp_miss+5;
end;

if es_naics_eso1997_miss=1 then do;
  es_naics_eso1997=mode_es_naics_eso1997_emp;
  es_naics_eso1997_miss=mode_es_naics_eso1997_emp_miss+5;
end;
if es_naics_eso2002_miss=1 then do;
  es_naics_eso2002=mode_es_naics_eso2002_emp;
  es_naics_eso2002_miss=mode_es_naics_eso2002_emp_miss+5;
end;

```

COUNTY .

```

if es_county_miss=1 and mode_es_county_emp_miss~=1 then do;
  es_county=mode_es_county_emp;
  es_county_miss=mode_es_county_emp_miss+5;
end;

```

OWNER_CODE .

```

if es_owner_code_miss=1 and mode_es_owner_code_emp_miss~=1 then do;
  es_owner_code=mode_es_owner_code_emp;
  es_owner_code_miss=mode_es_owner_code_emp_miss+5;
end;

```

EIN .

```

if es_ein_miss=1 and mode_es_ein_emp_miss~1 then do;
  es_ein=mode_es_ein_emp;
  es_ein_miss=mode_es_ein_emp_miss+5;
end;
run;

```

```

/* diagnostics moved to ecfdiag_25.sas */

```

1.5.2.26 library/sasprogs/26_ecf.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/26_ecf.sas >--+ */
/*----- LEGACY NAME: 26_seinunit_yq_chars.sas -----*/

```

Bring all the data together, apply variable labels, and perform some minor clean-up.

Programs 18 through 25 have been focused on filling industry and location missing values. The results of this effort are merged back onto the employment and earnings variables from step 12.

Variable lengths are set to minimize storage space

Variable labels are defined.

Create new aggregated industry classifications based on the detailed 4 digit SIC and 6 digit NAICS

Perform final clean up on ownership code and some of the geography variables.

```

/*****
/* REQUIRES-DATA: ss_employer_char_unit2.sas7bdat, seinunit_long_25.sas7bdat */
/* PROVIDES-DATA: ss_employer_char_unit3.sas7bdat */
*****/

title1 "Put Everything together";
title2 "Program Location: &curdir./26_sein_yq_chars.sas";

```

PUT IT ALL TOGETHER .

```

data INTERWRK.&state._employer_char_unit3();
  merge INTERWRK.&state._employer_char_unit2
        (in=_mvar1
          drop=_merge)
        INTERWRK.seinunit_long_25
        (in=_mvar2
          drop=merge best_emp1 best_emp2 best_emp3
                    EMP any_valid continue continue1
                    employ pct_emp yr_qtr found_ind_temp
                    lookup_temp low_limit naics1997_impute
                    naics2002_impute sic_impute
                    sum_n02 sum_n97 sum_sic uniform0-uniform3
                    up_limit
                    miss_naics1997_temp miss_naics_aux1997_temp
                    miss_naics_ldb1997_temp
                    es_sic_clean es_sic_temp sic_change
                    es_naics1997_clean es_naics1997_temp
                    es_naics2002_clean es_naics2002_temp
                    es_naics_aux1997_clean es_naics_aux1997_temp
                    es_naics_aux2002_clean es_naics_aux2002_temp
                    es_naics_ldb1997_clean es_naics_ldb1997_temp
                    es_naics_ldb2002_clean es_naics_ldb2002_temp
                    es_naics_imp1997_clean es_naics_imp1997_temp
                    es_naics_imp2002_clean
                    es_naics_fnl1997_temp
          );
  by sein year quarter seinunit;

```

Set the final lengths here. All fuzz variables are length 8.

```
length
  es_sic_2 $2
  es_sic_3 $3
  es_naics_fnl1997_2 $2
  es_naics_fnl1997_3 $3
  es_naics_fnl2002_2 $2
  es_naics_fnl2002_3 $3
  es_county_flag 3
  es_county_miss 3
  es_ein_flag 3
  es_ein_miss 3
  es_naics1997_flag 3
  es_naics1997_miss 3
  es_naics2002_flag 3
  es_naics2002_miss 3
  es_naics_aux1997_flag 3
  es_naics_aux1997_miss 3
  es_naics_aux2002_flag 3
  es_naics_aux2002_miss 3
  es_naics_ldb1997_flag 3
  es_naics_ldb1997_miss 3
  es_naics_ldb2002_flag 3
  es_naics_ldb2002_miss 3
  es_naics_eso1997_miss 3
  es_naics_eso2002_miss 3
  es_naics_fnl1997_miss 3
  es_naics_fnl2002_miss 3
  es_sic_valid 3
  es_naics1997_valid 3
  es_naics2002_valid 3
  es_naics_aux1997_valid 3
  es_naics_aux2002_valid 3
  es_naics_ldb1997_valid 3
  es_naics_ldb2002_valid 3
  es_owner_code_flag 3
  es_owner_code_miss 3
  es_sic_flag 3
  es_sic_miss 3
```

Mode variables.

```
mode_es_county_emp_flag 3
mode_es_county_emp_miss 3
mode_es_county_flag 3
mode_es_county_miss 3
mode_es_ein_emp_flag 3
mode_es_ein_emp_miss 3
mode_es_ein_flag 3
mode_es_ein_miss 3
mode_es_naics_eso1997_emp_flag 3
mode_es_naics_eso1997_emp_miss 3
mode_es_naics_eso1997_flag 3
mode_es_naics_eso1997_miss 3
mode_es_naics_eso2002_emp_flag 3
mode_es_naics_eso2002_emp_miss 3
mode_es_naics_eso2002_flag 3
mode_es_naics_eso2002_miss 3
mode_es_naics_fnl1997_emp_flag 3
mode_es_naics_fnl1997_emp_miss 3
mode_es_naics_fnl1997_flag 3
mode_es_naics_fnl1997_miss 3
mode_es_naics_fnl2002_emp_flag 3
mode_es_naics_fnl2002_emp_miss 3
mode_es_naics_fnl2002_flag 3
mode_es_naics_fnl2002_miss 3
mode_es_owner_code_emp_flag 3
mode_es_owner_code_emp_miss 3
mode_es_owner_code_flag 3
mode_es_owner_code_miss 3
mode_es_sic_emp_flag 3
mode_es_sic_emp_miss 3
mode_es_sic_flag 3
mode_es_sic_miss 3
mode_leg_state_emp_flag 3
mode_leg_state_emp_miss 3
mode_leg_state_flag 3
mode_leg_state_miss 3
mode_leg_county_emp_flag 3
mode_leg_county_emp_miss 3
mode_leg_county_flag 3
mode_leg_county_miss 3
mode_leg_msapmsa_emp_flag 3
mode_leg_msapmsa_emp_miss 3
mode_leg_msapmsa_flag 3
mode_leg_msapmsa_miss 3
mode_leg_subctygeo_emp_flag 3
mode_leg_subctygeo_emp_miss 3
mode_leg_subctygeo_flag 3
mode_leg_subctygeo_miss 3
mode_leg_wib_emp_flag 3
mode_leg_wib_emp_miss 3
mode_leg_wib_flag 3
mode_leg_wib_miss 3
;
```

Labels Go Here.

```
label yr_qtr='Continuous Time YEAR QUARTER';
label ES_STATE='ES202 FIPS State SS';
label ES_COUNTY='Cleaned ES202 FIPS County CCC';
label ES_EIN='Cleaned EIN';
label ES_SIC='Cleaned SIC Code IIII';
label ES_SIC_DIV='Cleaned SIC Division I';
label ES_SIC_2='Cleaned SIC Code II';
label ES_SIC_3='Cleaned SIC Code III';
label ES_NAICS1997="Cleaned 1997 NAICS Code NNNNNN";
label ES_NAICS2002="Cleaned 1997 NAICS Code NNNNNN";
label ES_NAICS_AUX1997="Cleaned 1997 NAICS Code NNNNNN";
label ES_NAICS_AUX2002="Cleaned 2002 NAICS Code NNNNNN";
label ES_NAICS_LDB1997="Cleaned 1997 NAICS Code NNNNNN";
label ES_NAICS_LDB2002="Cleaned 2002 NAICS Code NNNNNN";
label ES_NAICS_ES01997="ES202 ONLY 1997 NAICS Code NNNNNN";
label ES_NAICS_ES02002="ES202 ONLY 2002 NAICS Code NNNNNN";
label ES_NAICS_IMP1997="SIC IMP 1997 NAICS Code NNNNNN";
label ES_NAICS_IMP2002="SIC IMP 2002 NAICS Code NNNNNN";
label ES_NAICS_FNL1997="Final 1997 NAICS Code NNNNNN";
label ES_NAICS_FNL2002="Final 2002 NAICS Code NNNNNN";
label ES_NAICS_FNL1997_2="Final 1997 NAICS Code NN";
label ES_NAICS_FNL2002_2="Final 1997 NAICS Code NN";
label ES_NAICS_FNL1997_3="Final 1997 NAICS Code NNN";
label ES_NAICS_FNL2002_3="Final 1997 NAICS Code NNN";
label ES_NAICS_FNL1997_4="Final 1997 NAICS Code NNNN";
label ES_NAICS_FNL2002_4="Final 1997 NAICS Code NNNN";
label ES_NAICS_FNL1997_5="Final 1997 NAICS Code NNNNN";
label ES_NAICS_FNL2002_5="Final 1997 NAICS Code NNNNN";
label ES_OWNER_CODE="Cleaned OWNER_CODE 0";
```

```

label MODE_ES_COUNTY="Unit Mode Cleaned County";
label MODE_ES_COUNTY_EMP="Emp Mode Cleaned County";
label MODE_ES_EIN="Unit Mode Cleaned EIN";
label MODE_ES_EIN_EMP="Emp Mode Cleaned EIN";
label MODE_ES_NAICS_ESO1997="Unit Mode Cleaned NAICS 1997";
label MODE_ES_NAICS_ESO1997_EMP="Emp Mode Cleaned NAICS 1997";
label MODE_ES_NAICS_ESO2002="Unit Mode Cleaned NAICS 2002";
label MODE_ES_NAICS_ESO2002_EMP="Emp Mode Cleaned NAICS 2002";
label MODE_ES_NAICS_FNL1997="Unit Mode Cleaned NAICS 1997";
label MODE_ES_NAICS_FNL1997_EMP="Emp Mode Cleaned NAICS 1997";
label MODE_ES_NAICS_FNL2002="Unit Mode Cleaned NAICS 2002";
label MODE_ES_NAICS_FNL2002_EMP="Emp Mode Cleaned NAICS 2002";
label MODE_ES_OWNER_CODE="Unit Mode Cleaned OWNER_CODE";
label MODE_ES_OWNER_CODE_EMP="Emp Mode Cleaned OWNER_CODE";
label MODE_ES_SIC="Unit Mode Cleaned SIC";
label MODE_ES_SIC_EMP="Emp Mode Cleaned SIC";

label MODE_LEG_MSAPMSA="Unit Mode Cleaned GEO MSAPMSA";
label MODE_LEG_MSAPMSA_EMP="Emp Mode Cleaned GEO MSAPMSA";
label MODE_LEG_WIB="Unit Mode Cleaned GEO WIB";
label MODE_LEG_WIB_EMP="Emp Mode Cleaned GEO WIB";
label MODE_LEG_STATE="Unit Mode Cleaned GEO STATE";
label MODE_LEG_STATE_EMP="Emp Mode Cleaned GEO STATE";
label MODE_LEG_COUNTY="Unit Mode Cleaned GEO COUNTY";
label MODE_LEG_COUNTY_EMP="Emp Mode Cleaned GEO COUNTY";
label MODE_LEG_SUBCTYGEO="Unit Mode Cleaned GEO COUNTY";
label MODE_LEG_SUBCTYGEO_EMP="Emp Mode Cleaned GEO COUNTY";

label LEG_COUNTY='Cleaned GEO FIPS County CCC';
label LEG_STATE='Cleaned GEO State SS';
label NUM_ESTABS='Number of Establishments';

label best_emp1='Best UI/202 Employment Month 1';
label best_emp2='Best UI/202 Employment Month 2';
label best_emp3='Best UI/202 Employment Month 3';
label best_flag='Source of best_data';
label best_wages='Best UI/202 Wages';

label county='Original ES202 County';
label ein='Original ES202 EIN';
label ein_bad='Letters a-z,A-Z in EIN';
label ein_defect='Problem with EIN';
label valid_ein='EIN in known IRD';

label emp1_UI='Best SEIN UI Employment';
label emp2_UI='Best SEIN UI Employment';
label emp3_UI='Best SEIN UI Employment';
label wages_UI='SEIN UI Wages';
label emp1_month1='Original ES202 Employment Month 1';
label emp1_month2='Original ES202 Employment Month 2';
label emp1_month3='Original ES202 Employment Month 3';

```

```

label master_empl_month1_flg='Stored Master Record Flag';
label master_empl_month2_flg='Stored Master Record Flag';
label master_empl_month3_flg='Stored Master Record Flag';
label master_multi_unit_code='Stored Master Multi Code';
label master_total_wages_flg='Stored Master Record Flag';

label es_county_flag='Quarters Away County data found';
label es_ein_flag='Quarters Away EIN data found';
label es_naics1997_flag='Quarters Away NAICS data found';
label es_naics2002_flag='Quarters Away NAICS data found';
label es_naics_aux1997_flag='Quarters Away NAICS data found';
label es_naics_aux2002_flag='Quarters Away NAICS data found';
label es_naics_ldb1997_flag='Quarters Away NAICS data found';
label es_naics_ldb2002_flag='Quarters Away NAICS data found';
label es_owner_code_flag='Quarters Away OWNER_CODE data found';
label es_sic_flag='Quarters Away SIC data found';

label es_sic_valid='Seinunit has some SIC info';
label es_naics1997_valid='Seinunit has some NAICS info';
label es_naics2002_valid='Seinunit has some NAICS info';
label es_naics_aux1997_valid='Seinunit has some NAICS info';
label es_naics_aux2002_valid='Seinunit has some NAICS info';
label es_naics_ldb1997_valid='Seinunit has some NAICS info';
label es_naics_ldb2002_valid='Seinunit has some NAICS info';

label es_county_miss='0=ok,1=not found,2+found off qtr';
label es_ein_miss='0=ok,1=not found,2+found off qtr';
label es_naics1997_miss='0=ok,1=not found,2+found off qtr';
label es_naics2002_miss='0=ok,1=not found,2+found off qtr';
label es_naics_aux1997_miss='0=ok,1=not found,2+found off qtr';
label es_naics_aux2002_miss='0=ok,1=not found,2+found off qtr';
label es_naics_ldb1997_miss='0=ok,1=not found,2+found off qtr';
label es_naics_ldb2002_miss='0=ok,1=not found,2+found off qtr';
label es_naics_eso1997_miss='0=ok,1=not found,2+found off qtr';
label es_naics_eso2002_miss='0=ok,1=not found,2+found off qtr';
label es_naics_imp1997_miss='0=ok,1=not found,2+found off qtr';
label es_naics_imp2002_miss='0=ok,1=not found,2+found off qtr';
label es_naics_fnl1997_miss='0=ok,1=not found,2+found off qtr';
label es_naics_fnl2002_miss='0=ok,1=not found,2+found off qtr';
label es_owner_code_miss='0=ok,1=not found,2+found off qtr';
label es_sic_miss='0=ok,1=not found,2+found off qtr';

```

```

label es_sic_src="Source of Ind Code";
label es_naics1997_src="Source of Ind Code";
label es_naics2002_src="Source of Ind Code";
label es_naics_aux1997_src="Source of Ind Code";
label es_naics_aux2002_src="Source of Ind Code";
label es_naics_ldb1997_src="Source of Ind Code";
label es_naics_ldb2002_src="Source of Ind Code";
label es_naics_eso1997_src="Source of Ind Code";
label es_naics_eso2002_src="Source of Ind Code";
label es_naics_imp1997_src="Source of Ind Code";
label es_naics_imp2002_src="Source of Ind Code";
label es_naics_fnl1997_src="Source of Ind Code";
label es_naics_fnl2002_src="Source of Ind Code";

label ever_202='SEIN ever on 202';
label ever_UI='SEIN ever on UI';

label ever_emp1="MULTI ever has ES202 month 1 employment";
label ever_emp2="MULTI ever has ES202 month 2 employment";
label ever_emp3="MULTI ever has ES202 month 3 employment";
label ever_multi="SEIN ever multi unit";
label ever_wages="MULTI ever ES202 wages";

label in_202="SEIN in ES202";
label in_UI="SEIN in UI";

label mode_es_county_emp_flag      =      "Quarters Away Data Found";
label mode_es_county_emp_miss     =      "Missing Value" ;
label mode_es_county_flag         =      "Quarters Away Data Found";
label mode_es_county_miss        =      "Missing Value" ;
label mode_es_ein_emp_flag        =      "Quarters Away Data Found";
label mode_es_ein_emp_miss       =      "Missing Value" ;
label mode_es_ein_flag            =      "Quarters Away Data Found";
label mode_es_ein_miss           =      "Missing Value" ;
label mode_es_naics_eso1997_emp_flag = "Quarters Away Data Found";
label mode_es_naics_eso1997_emp_miss = "Missing Value" ;
label mode_es_naics_eso1997_flag  =      "Quarters Away Data Found";
label mode_es_naics_eso1997_miss  =      "Missing Value" ;
label mode_es_naics_eso2002_emp_flag = "Quarters Away Data Found";
label mode_es_naics_eso2002_emp_miss = "Missing Value" ;
label mode_es_naics_eso2002_flag  =      "Quarters Away Data Found";
label mode_es_naics_eso2002_miss  =      "Missing Value" ;
label mode_es_naics_fnl1997_emp_flag = "Quarters Away Data Found";
label mode_es_naics_fnl1997_emp_miss = "Missing Value" ;
label mode_es_naics_fnl1997_flag  =      "Quarters Away Data Found";
label mode_es_naics_fnl1997_miss  =      "Missing Value" ;
label mode_es_naics_fnl2002_emp_flag = "Quarters Away Data Found";
label mode_es_naics_fnl2002_emp_miss = "Missing Value" ;
label mode_es_naics_fnl2002_flag  =      "Quarters Away Data Found";
label mode_es_naics_fnl2002_miss  =      "Missing Value" ;
label mode_es_owner_code_emp_flag  =      "Quarters Away Data Found";
label mode_es_owner_code_emp_miss  =      "Missing Value" ;
label mode_es_owner_code_flag     =      "Quarters Away Data Found";
label mode_es_owner_code_miss     =      "Missing Value" ;
label mode_es_sic_emp_flag         =      "Quarters Away Data Found";
label mode_es_sic_emp_miss        =      "Missing Value" ;
label mode_es_sic_flag            =      "Quarters Away Data Found";
label mode_es_sic_miss            =      "Missing Value" ;

```

```

label mode_leg_state_emp_flag      =      "Quarters Away Data Found";
label mode_leg_state_emp_miss     =      "Missing Value" ;
label mode_leg_state_flag         =      "Quarters Away Data Found";
label mode_leg_state_miss        =      "Missing Value" ;
label mode_leg_county_emp_flag    =      "Quarters Away Data Found";
label mode_leg_county_emp_miss   =      "Missing Value" ;
label mode_leg_county_flag       =      "Quarters Away Data Found";
label mode_leg_county_miss       =      "Missing Value" ;
label mode_leg_msapmsa_emp_flag   =      "Quarters Away Data Found";
label mode_leg_msapmsa_emp_miss  =      "Missing Value" ;
label mode_leg_msapmsa_flag      =      "Quarters Away Data Found";
label mode_leg_msapmsa_miss      =      "Missing Value" ;
label mode_leg_subctygeo_emp_flag =      "Quarters Away Data Found";
label mode_leg_subctygeo_emp_miss =      "Missing Value" ;
label mode_leg_subctygeo_flag    =      "Quarters Away Data Found";
label mode_leg_subctygeo_miss    =      "Missing Value" ;
label mode_leg_wib_emp_flag       =      "Quarters Away Data Found";
label mode_leg_wib_emp_miss      =      "Missing Value" ;
label mode_leg_wib_flag          =      "Quarters Away Data Found";
label mode_leg_wib_miss          =      "Missing Value" ;

label multi_first_quarter='First Quarter SEIN on 202';
label multi_first_year="First Year SEIN on 202";
label multi_unit='SEIN w/2+ records on 202';
label naics1997='Original NAICS 1997 Code';
label naics2002='Original NAICS 2002 Code';
label naics_aux1997='Original NAICS AUX 1997 Code';
label naics_aux2002='Original NAICS AUX 2002 Code';
label naics_ldb1997='Original NAICS LDB 1997 Code';
label naics_ldb2002='Original NAICS LDB 2002 Code';

label owner_code='Original Owner Code';
label payroll='Original UI Payroll Info W1';

label sein_best_emp1='SEIN Best UI/202 Month 1, Employment';
label sein_best_emp2='SEIN Best UI/202 Month 2, Employment';
label sein_best_emp3='SEIN Best UI/202 Month 3, Employment';
label sein_best_wages='SEIN Best UI/202 Payroll';

label sein_emp1='SEIN 202 Employment Month 1';
label sein_emp2='SEIN 202 Employment Month 2';
label sein_emp3='SEIN 202 Employment Month 3';
label sein_wages='SEIN 202 Wages';

label seinsize_m='UI Employment M';
label seinsize_b='UI Employment B';
label seinsize_e='UI Employment E';

label seinunit_bad='SEINUNIT data non-numeric';
label seinunit_type='0 if seinunit=00000';

label sic='Original ES202 SIC';

label sic_invalid='SIC Code not Valid';
label naics_1997_invalid='NAICS Code not Valid';
label naics_2002_invalid='NAICS Code not Valid';
label naics_aux_1997_invalid='NAICS Code not Valid';
label naics_aux_2002_invalid='NAICS Code not Valid';
label naics_ldb_1997_invalid='NAICS Code not Valid';
label naics_ldb_2002_invalid='NAICS Code not Valid';

label special_handle='candidate for structure fix';
label structure_fix='Multiunit Imputed Record Structure';
label total_wages='Original ES202 wages';

```

Some lengths.

```

length ES_SIC_DIV $1 ES_SIC_2 $2 ES_SIC_3 $3
      ES_NAICS_FNL1997_2 ES_NAICS_FNL2002_2 $2
      ES_NAICS_FNL2002_3 ES_NAICS_FNL2002_3 $3
      ES_NAICS_FNL2002_4 ES_NAICS_FNL2002_4 $4
      ES_NAICS_FNL2002_5 ES_NAICS_FNL2002_5 $5;

if _mvar1=1 and _mvar2=1 then _merge=3;
if _mvar1=1 and _mvar2=0 then _merge=1;
if _mvar1=0 and _mvar2=1 then _merge=2;

```

Create the SIC and NAICS Variables .

```

ES_SIC_2=substr(ES_SIC,1,2);
ES_SIC_3=substr(ES_SIC,1,3);
%conv2div(ES_SIC_2,ES_SIC_DIV);

ES_NAICS_FNL1997_2=substr(ES_NAICS_FNL1997,1,2);
ES_NAICS_FNL1997_3=substr(ES_NAICS_FNL1997,1,3);
ES_NAICS_FNL1997_4=substr(ES_NAICS_FNL1997,1,4);
ES_NAICS_FNL1997_5=substr(ES_NAICS_FNL1997,1,5);
ES_NAICS_FNL2002_2=substr(ES_NAICS_FNL2002,1,2);
ES_NAICS_FNL2002_3=substr(ES_NAICS_FNL2002,1,3);
ES_NAICS_FNL2002_4=substr(ES_NAICS_FNL2002,1,4);
ES_NAICS_FNL2002_5=substr(ES_NAICS_FNL2002,1,5);

```

Fix the Owner Code .

```

if es_owner_code="9" then do;
  es_owner_code='5';
  es_owner_code_miss=12;
end;

if mode_es_owner_code="9" then do;
  mode_es_owner_code='5';
  mode_es_owner_code_miss=12;
end;

if mode_es_owner_code_emp="9" then do;
  mode_es_owner_code_emp='5';
  mode_es_owner_code_emp_miss=12;
end;

```

Clean up the mode Geography Variables.

```

if mode_leg_wib_emp_miss=1 then do;
  mode_leg_wib_emp=mode_leg_wib;
  mode_leg_wib_emp_miss=5;
end;

if mode_leg_msapmsa_emp_miss=1 then do;
  mode_leg_msapmsa_emp=mode_leg_msapmsa;
  mode_leg_msapmsa_emp_miss=5;
end;

if mode_leg_state_emp_miss=1 then do;
  mode_leg_state_emp=mode_leg_state;
  mode_leg_state_emp_miss=5;
end;

if mode_leg_county_emp_miss=1 then do;
  mode_leg_county_emp=mode_leg_county;
  mode_leg_county_emp_miss=5;
end;

if mode_leg_subctygeo_emp_miss=1 then do;
  mode_leg_subctygeo_emp=mode_leg_subctygeo;
  mode_leg_subctygeo_emp_miss=5;
end;

run;

```

Diagnostics moved to ecfdiag_26.sas

1.5.2.27 library/sasprogs/27_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/sasprogs/27_ecf.sas >---+ */
/*----- LEGACY NAME: 27_weight_calc_01.sas -----*/

```

Creation of the weights, step 1.

First, the sample over which the weights will be positive (and therefore which units will be included in any calculations) is determined. If a unit has ever appeared on the ES202 and is in the UI then it is eligible for a positive weight.

Additional restrictions are imposed for the actual calculation of the overall adjustment. The overall adjustment factor is calculated for private firms (es_owner_code=5 and the first digit of es_sic not equal to 9). This adjustment is later applied to public firms as well.

The second component of the weights is based on the difference between UI and ES202 employment at the SEIN level. These differences are calculated and trimmed if necessary. The bounds for the trim at the time of this documents creation are .7 and 1.3, and are hard-coded into the parameters file.

See the technical paper on the weights for more information.

```

/*****
/* REQUIRES-SAS: runtime_ecf.sas */
/* REQUIRES-DATA: ss_employer_char_unit3.sas7bdat */
/* PROVIDES-DATA: weights_sein_27.sas7bdat */
*****/

* %include "runtime_ecf.sas";

*options obs=100000;

title1 "Calculate the Weight from the ECF";
title2 "Program Location: &curdir./27_weight_calc_01.sas";

```

Get data off the ECF.

```
data weights_sein_01
  (keep=sein year quarter sein_best_emp1 emp1_UI in_UI in_202
   ever_UI ever_202
   sein_best_emp1_weight sein_best_emp1_adjust
   count_inscope get_weight seinsize_b
  );
set INTERWRK.&state._employer_char_unit3
  (keep=sein year quarter seinunit es_owner_code es_sic leg_state
   best_emp1 sein_best_emp1 emp1_UI in_UI in_202 ever_UI
   ever_202 best_flag seinsize_b
  );
by sein year quarter;

if first.quarter then do;
  sein_best_emp1_adjust=0;
  sein_best_emp1_weight=0;
  count_inscope=0;
end;

if es_owner_code='5' and substr(es_sic,1,1)~='9' and leg_state="&stfips"
then do;
  count_inscope+1;
  sein_best_emp1_weight+best_emp1;
end;

sein_best_emp1_adjust+best_emp1;

if last.quarter then do;
  get_weight=0;
  if ever_202=1 and in_UI=1 then get_weight=1;
  if sein_best_emp1_weight=0 and count_inscope>0 then
sein_best_emp1_weight=emp1_UI;
  if sein_best_emp1_adjust=0 then sein_best_emp1_adjust=emp1_UI;
  output weights_sein_01;
end;
run;
```

Diagnostics.

```
proc freq data=weights_sein_01;
  tables year*get_weight;
  tables year*ever_UI;
  tables ever_UI*in_UI;
  tables ever_202*in_UI;
  tables ever_202*in_UI*get_weight;
run;
```

Calculate W2 and Trim the result .

```

data weights_sein_02;
  set weights_sein_01;

  length rectype $7;

  if in_ui=1 and in_202=1 then rectype="ESandUI";
  if in_ui=1 and in_202=0 then rectype="UIonly";
  if in_ui=0 and in_202=1 then rectype="ESonly";

  if get_weight=1 then do;
    w2=sein_best_emp1_adjust/emp1_UI;
    w2_trim=w2;
    if w2>&utrim then w2_trim=&utrim;
    if w2<&ltrim then w2_trim=&ltrim;
    emp1_w2=w2*emp1_UI;
    emp1_w2_trim=w2_trim*emp1_UI;
    b_w2=w2*seinsize_b;
    b_w2_trim=w2_trim*seinsize_b;
  end;

run;

```

Diagnostics moved to ecfdiag_27.sas

Sort the data in preparation for the next step .

```

proc sort data=weights_sein_02 out=INTERWRK.weights_sein_27;
  by year quarter;
run;

```

1.5.2.28 library/sasprogs/28_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/sasprogs/28_ecf.sas >--- */
/*----- LEGACY NAME: 28_weight_calc_02.sas -----*/

```

Creation of the weights, step 2.

Calculate the UI totals necessary to create the overall adjustment.

The UI totals are combined with the BLS totals and the SEIN level adjustment to create the final weight.

See the technical paper on the weights for more information.

```

/*****
***/
/* REQUIRES-DATA: BLS totals data bls_us_totals.sas7bdat,
weights_sein_27.sas7bdat*/
/*
      INTERWRK.ecf_lehdtotals_10
*/
/* REQUIRES-DATA: INPUTS.ehf_&state._controltotals
*/
/* PROVIDES-DATA: weights_totals_ss_28.sas7bdat, weights_sein_28.sas7bdat
*/
/*
      ss_employer_char_unit4.sas7bdat
*/
/*****
***/

title1 "Calculate the Weight from the ECF";
title2 "Program Location: &curdir./28_weight_calc_02.sas";

```

The code combining the BLS and UI totals was moved to EHF/3.1.10 in March 2004.

Construct step1 as in old days, except that we use EHF_controltotals here.

```

data step1(keep=leg_state year quarter bls_emp_month1 state_name yr_qtr);
  set INPUTS.ehf_&state._controltotals;
  run;

```

Calculate totals from the ECF.

```

proc means data=INTERWRK.weights_sein_27(where=(get_weight=1 and
count_inscope>0)) n sum ;
  title1 "sums for get_weight=1";
  class year quarter;
  var sein_best_emp1 sein_best_emp1_adjust sein_best_emp1_weight
  emp1_w2 emp1_w2_trim emp1_UI seinsize_b;
  output out=step2 (drop=)
  sum(sein_best_emp1 sein_best_emp1_adjust sein_best_emp1_weight
  emp1_w2 emp1_w2_trim emp1_UI seinsize_b b_w2 b_w2_trim)=
  ecf_best_emp1 ecf_best_emp1_adjust ecf_best_emp1_weight
  ecf_emp1_w2 ecf_emp1_w2_trim ecf_emp1_UI ecf_b ecf_b_w2 ecf_b_w2_trim;
run;

```

Diagnostics

```

proc print;
  title1 "TOTALS from the ECF";
run;

```

Merge the BLS totals with the ECF totals:

- w0=target adjustment for B
- w1=target adjustment for emp1_UI

```

data INTERWRK.weight_totals_&state._28(drop=_TYPE_ rename=( _FREQ_=NUM_SEIN));
  merge step1 step2(where=( _TYPE_=3));
  by year quarter;

  if ecf_b_w2>0 then w0=bls_emp_month1/ecf_b_w2;
  if ecf_b_w2_trim>0 then w0_trim=bls_emp_month1/ecf_b_w2_trim;
  w1=bls_emp_month1/ecf_best_emp1_weight;
  w1_trim=bls_emp_month1/ecf_emp1_w2_trim;
run;

```

Diagnostics moved to ecfdiag_28.sas.

Merge on the totals data .

```

data INTERWRK.weights_sein_28;
  merge INTERWRK.weights_sein_27(in=a) INTERWRK.weight_totals_&state._28(in=b);
  by year quarter;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;

  qwi_unit_weight=0;
  if get_weight=1 and w0_trim>0 then qwi_unit_weight=w0_trim*w2_trim;

  weight=0;
  if get_weight=1 then weight=w1_trim*w2_trim;

  weight_notrim=0;
  if get_weight=1 then weight_notrim=w1*w2;

  emp1_UI_weight=weight*emp1_UI;
  emp1_UI_notrim=weight_notrim*emp1_UI;
  emp1_UI_adjust=qwi_unit_weight*emp1_UI;
  b_adjust=qwi_unit_weight*seinsize_b;

  label qwi_unit_weight='Weight sum(B_UI)=sum(month1_BLS)';
run;

```

Diagnostics

```

proc freq;
  tables merge;
  tables yr_qtr*get_weight;
run;

proc sort data=INTERWRK.weights_sein_28;
  by sein year quarter;
run;

```

Merge the weights on before adding the fuzz.

```

data INTERWRK.&state._employer_char_unit4;
  merge
    INTERWRK.&state._employer_char_unit3(in=a drop=_merge)
    INTERWRK.weights_sein_28(in=b keep=sein year quarter qwi_unit_weight)
  ;
  by sein year quarter;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;
  if merge=1 or merge=3 then output;
run;

```

Diagnostics moved to ecfdiag_28.sas

Final sort

```

proc sort data=INTERWRK.&state._employer_char_unit4;
  by sein seinunit year quarter;
run;

```

1.5.2.29 library/sasprogs/29_ecf.sas

```

/* +---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* +---< Location: /programs/production/dev1/current/ecf >---+ */
/* +---< File: library/sasprogs/29_ecf.sas >---+ */
/*----- LEGACY NAME: 29_seinunit_fuzz.sas -----*/

```

This program attaches fuzz factors to the ECF, and creates the final ECF SEINUNIT file.

An SEIN and SEINUNIT has a constant fuzz factor over time. The two are related such that if the SEIN fuzz factor is less than one then all SEINUNIT fuzz factors for that firm are also less than one. A fuzz factor, once assigned, never changes.

When applying fuzz factors, if the SEIN SEINUNIT was previously part of the ECF then the existing fuzz factor is used. If not, then the fuzz factor is calculated using two different distributions and stored in a temporary file for subsequent use.

The fuzz factor for each SEIN and SEIN SEINUNIT are stored in separate files in the fuzz directory and should NEVER be removed. The first time this program is run for a state a new empty file is created. This file will be filled later.

```

/*****
/* REQUIRES-DATA: ss_employer_char_unit_4.sas7bdat */
/* ss_master_sein_fuzz.sas7bdat */
/* ss_master_seinunit_fuzz.sas7bdat */
/* PROVIDES-DATA: ecf_seinunit_ss.sas7bdat */
/* new_sein_fuzz_29.sas7bdat */
/* new_seinunit_fuzz_29.sas7bdat */
*****/

title1 "Fuzz it all";
title2 "Program Location: &curdir./29_seinunit_fuzz.sas";

```

First check to see if the fuzz files exist We don't want to overwrite any fuzz factors!! If the files do not exist we create empty ones with the correct structure in INTERWRK. If they do exist, we copy them from INPUTS to INTERWRK.

```
%fzexist(INPUTS.ecf_&state._sein_fuzz, INPUTS.ecf_&state._seinunit_fuzz);
```

Read in the (confidential) shape parameters.

```
*** Distribution Shape ***;
data _null_;
set INPUTS.parms_us_fuzz;
call symput(name,trim(left(value)));
put "SHAPE PARAMETERS: " name= value=;
run;
```

We'll need two passes through the data. This pass gets the direction(SEIN).

```
/*BEGINCONFIDENTIAL====*/

data pass1 INTERWRK.new_sein_fuzz_29(keep=sein UNIFORM_SEIN BETA_SEIN
RAMP_SEIN);
merge INTERWRK.&state._employer_char_unit4(in=a drop=merge)
INTERWRK.&state._master_sein_fuzz(in=b);
by sein;

if a=1 and b=1 then merge=3;
if a=1 and b=0 then merge=1;
if a=0 and b=1 then merge=2;

retain UNIFORM_SEIN BETA_SEIN RAMP_SEIN;

/*====CONFIDENTIALEND*/
```

Get the SEIN fuzz factor . Use the old one if available, If not create a new one.

```
if first.sein then do;
if merge=1 then do;
```

Generate new fuzz factor.

```
/*BEGINCONFIDENTIAL====*/
UNIFORM_SEIN=ranuni(389758679);
%addnoise(RAMP_SEIN,UNIFORM_SEIN,ramp,&fuzz_lb_cat1,&fuzz_ub_cat1);

%addnoise(BETA_SEIN,UNIFORM_SEIN,beta,&fuzz_lb_cat1,&fuzz_ub_cat1,&fuzz_q,&fuzz_
p);
/*====CONFIDENTIALEND*/
output INTERWRK.new_sein_fuzz_29;
end;
end;

if merge=1 or merge=3 then output pass1;
run;
```

Diagnostics. Allows to check how many new fuzz factors were created. Should this maybe go to QA?

```
proc freq data=pass1;
tables merge;
run;
```

The second pass through creates the final ECF SEINUNIT file

```

data OUTPUTS.ecf_&state._seinunit
  (sortedby=sein seinunit year quarter
  index=(
    sein sein_seinunit_year_quarter=(sein seinunit year quarter)
    sein_year_quarter_seinunit=(sein year quarter seinunit)
    sein_year_quarter=(sein year quarter)
  )
  compress=binary
  drop=merge
)
merge_check(keep=merge)
/*BEGINCONFIDENTIAL====*/
  INTERWRK.new_seinunit_fuzz_29(keep=sein seinunit UNIFORM_SEINUNIT
  BETA_SEINUNIT RAMP_SEINUNIT)
;
merge
  pass1(in=a drop=merge)
  INTERWRK.&state._master_seinunit_fuzz(in=b);
  by sein seinunit;

  if a=1 and b=1 then merge=3;
  if a=1 and b=0 then merge=1;
  if a=0 and b=1 then merge=2;

  *** addnoise2(outvar,seed,distrib,lower,upper,beta1,beta2);

  retain UNIFORM_SEINUNIT BETA_SEINUNIT RAMP_SEINUNIT;
/*CONFIDENTIALEND*/

```

Get the SEINUNIT fuzz factor. Use the old one if available, If not create a new one.

```

  if first.seinunit then do;
    if merge=1 then do;
/*BEGINCONFIDENTIAL====*/
      UNIFORM_SEINUNIT=ranuni(8121249);
      if UNIFORM_SEIN>.5 then do;

%addnoise(RAMP_SEINUNIT,UNIFORM_SEINUNIT,ramp2,&fuzz_lb_cat1,&fuzz_ub_cat1);

%addnoise(BETA_SEINUNIT,UNIFORM_SEINUNIT,beta2,&fuzz_lb_cat1,&fuzz_ub_cat1,&fuzz
_q,&fuzz_p);
      end;
      else do;

%addnoise(RAMP_SEINUNIT,UNIFORM_SEINUNIT,ramp1,&fuzz_lb_cat1,&fuzz_ub_cat1);

%addnoise(BETA_SEINUNIT,UNIFORM_SEINUNIT,beta1,&fuzz_lb_cat1,&fuzz_ub_cat1,&fuzz
_q,&fuzz_p);
      end;
/*====CONFIDENTIALEND*/
      output INTERWRK.new_seinunit_fuzz_29;
      end;
    end;

    if merge=1 or merge=3 then do;
      output OUTPUTS.ecf_&state._seinunit;
      output merge_check;
    end;
  run;

```

Diagnostic output mostly moved to ecfdiag_29.sas.

```

proc freq data=merge_check;
  tables merge;
run;

```

1.5.2.30 library/sasprogs/30_ecf.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* ---< Location: /programs/production/dev1/current/ecf >--+ */
/* ---< File: library/sasprogs/30_ecf.sas >--+ */
/*----- LEGACY NAME: 30_sein_file.sas -----*/

```

This program outputs an SEIN level file.

Some data users only require information at the firm level. To make data access easier, a subset of the variables that do not vary within an SEIN YEAR QUARTER are output to a separate file. No aggregation is done - simply the first record within a quarter (i.e. from the first SEINUNIT) is output.

```

/*****
/* REQUIRES-DATA: OUTPUTS.ecf_seinunit_ss */
/* PROVIDES-DATA: OUTPUTS.ecf_sein_ss */
*****/

title1 "Create the SEIN File";
title2 "Program Location: &curdir./30_sein_file.sas";

data OUTPUTS.ecf_&state._sein
(
  sortedby=sein year quarter
  index=(sein sein_year_quarter=(sein year quarter))
  compress=binary
);

```

Keep only certain variables.

```

set OUTPUTS.ecf_&state._seinunit
  (keep=sein year quarter yr_qtr
/*BEGINCONFIDENTIAL=====*/
  beta_sein ramp_sein uniform_sein
/*=====CONFIDENTIALEND*/
  es_state
  mode_es_sic mode_es_sic_miss mode_es_sic_flag
  mode_es_sic_emp mode_es_sic_emp_miss mode_es_sic_emp_flag
  mode_es_naics_eso1997 mode_es_naics_eso1997_miss
    mode_es_naics_eso1997_flag
  mode_es_naics_eso1997_emp mode_es_naics_eso1997_emp_miss
    mode_es_naics_eso1997_emp_flag
  mode_es_naics_eso2002 mode_es_naics_eso2002_miss
    mode_es_naics_eso2002_flag
  mode_es_naics_eso2002_emp mode_es_naics_eso2002_emp_miss
    mode_es_naics_eso2002_emp_flag
  mode_es_naics_fnl1997 mode_es_naics_fnl1997_miss
    mode_es_naics_fnl1997_flag
  mode_es_naics_fnl1997_emp mode_es_naics_fnl1997_emp_miss
    mode_es_naics_fnl1997_emp_flag
  mode_es_naics_fnl2002 mode_es_naics_fnl2002_miss
    mode_es_naics_fnl2002_flag
  mode_es_naics_fnl2002_emp mode_es_naics_fnl2002_emp_miss
    mode_es_naics_fnl2002_emp_flag
  mode_es_county mode_es_county_miss mode_es_county_flag
  mode_es_county_emp mode_es_county_emp_miss
    mode_es_county_emp_flag
  mode_es_owner_code mode_es_owner_code_miss
    mode_es_owner_code_flag
  mode_es_owner_code_emp mode_es_owner_code_emp_miss
    mode_es_owner_code_emp_flag
  mode_es_ein mode_es_ein_miss mode_es_ein_flag
  mode_es_ein_emp mode_es_ein_emp_miss mode_es_ein_emp_flag
  mode_leg_wib mode_leg_wib_miss mode_leg_wib_flag
  mode_leg_wib_emp mode_leg_wib_emp_miss mode_leg_wib_emp_flag
  mode_leg_msapmsa mode_leg_msapmsa_miss mode_leg_msapmsa_flag
  mode_leg_msapmsa_emp mode_leg_msapmsa_emp_miss
    mode_leg_msapmsa_emp_flag
  mode_leg_state mode_leg_state_miss mode_leg_state_flag
  mode_leg_state_emp mode_leg_state_emp_miss
    mode_leg_state_emp_flag
  mode_leg_county mode_leg_county_miss mode_leg_county_flag
  mode_leg_county_emp mode_leg_county_emp_miss
    mode_leg_county_emp_flag
  mode_leg_subctygeo mode_leg_subctygeo_miss
    mode_leg_subctygeo_flag
  mode_leg_subctygeo_emp mode_leg_subctygeo_emp_miss
    mode_leg_subctygeo_emp_flag
  ever_202 ever_ui ever_emp1 ever_emp2 ever_emp3
  ever_multi ever_wages
  in_202 in_UI multi_first_quarter multi_first_year multi_unit
  sein_best_emp1 sein_best_emp2 sein_best_emp3 sein_best_wages
  payroll seinsize_m seinsize_e seinsize_b
  emp1_UI emp2_UI emp3_UI wages_UI
  sein_emp1 sein_emp2 sein_emp3 sein_wages num_estabs source
  qwi_unit_weight);

  by sein year quarter;
  if first.quarter then output;

run;

```

Diagnostics moved to ecfdiag_30.sas

1.5.2.31 library/sasprogs/31_ecf.sas

```
/* +---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* +---< Location: /programs/production/dev1/current/ecf >---+ */
/* +---< File: library/sasprogs/31_ecf.sas >---+ */
/*----- LEGACY NAME: 31_fuzz_update.sas -----*/
```

This program updates the master fuzz factor dataset. It used to update the file in place. In the production system, the file gets read from INPUT in step 29, gets processed in INTERWRK, and then written to OUTPUT. Rather than having a transaction file, there is a sequence of increasingly large files.

The first step is to calculate the largest update number on the fuzz factor dataset. The new additions to the fuzz file receive an update number one larger than the current max value, unless the file is empty and the number for all records is set to 0. [This may no longer be strictly necessary]

In addition to the update number, the date the fuzz factor was created is added to the master fuzz file. These two pieces allow for the auditing and tracking of all additions to the master fuzz file.

The new fuzz factors are merged onto the master fuzz factor file. If a new fuzz factor has the same SEIN or SEIN SEINUNIT as an existing record then processing is immediately halted, thus preventing any pollution of the existing fuzz factors.

```
/*-----**
**/
/* REQUIRES-DATA: new_sein_fuzz_29.sas7bdat new_seinunit_fuzz_29.sas7bdat
*/
/*          ss_master_sein_fuzz.sas7bdat,
ss_master_seinunit_fuzz.sas7bdat */
/* PROVIDES-DATA: ss_master_sein_fuzz.sas7bdat,
ss_master_seinunit_fuzz.sas7bdat */
/*-----**
**/

title1 "Update the Fuzz Files";
title2 "Program Location: &curdir./31_fuzz_update.sas";
```

Get the largest value of the update number variable .

```
%global cupdte;
%upnum(INTERWRK.&state._master_sein_fuzz,UPDATE_NUMBER_SEIN);
```

Do the actual update of the SEIN fuzz dataset .

```
data OUTPUTS.ecf_&state._sein_fuzz(drop=merge)
  INTERWRK.fuzz_merge_check1(keep=merge);
  merge
  INTERWRK.&state._master_sein_fuzz(in=a)
  INTERWRK.new_sein_fuzz_29(in=b);
  by sein;
```

Stata-style merge control.

```
if a=1 and b=1 then merge=3;
if a=1 and b=0 then merge=1;
if a=0 and b=1 then merge=2;
```

Records only on the original file are passed through untouched

```

if merge=1 then do;
  output OUTPUTS.ecf_&state._sein_fuzz;
end;

```

New records are added to the file, including a date stamp.

```

if merge=2 then do;
  DATE_SEIN_FUZZ=date();
  UPDATE_NUMBER_SEIN=&cupdte;
  output OUTPUTS.ecf_&state._sein_fuzz;
end;

```

Any matched records indicates an error. This should be improved on: output without abort, then flag QA.

```

output INTERWRK.fuzz_merge_check1;
if merge=3 then do;
  put "%upcase(error) Merge error in the fuzz factors";
  put "%upcase(error) Merge error in the fuzz factors";
  put "%upcase(error) Merge error in the fuzz factors";
  abortabend;
end;
run;

```

Diagnostics and QA on merge. Instructions: check the merge_check file. If no 3s, then OK, otherwise critical error. Most diagnostics moved to ecfdiag.31.sas

Get the largest value of the update number variable .

```

%let cupdte=;
%upnum(INTERWRK.&state._master_seinunit_fuzz,UPDATE_NUMBER_SEINUNIT);

```

Do the actual update of the SEINUNIT fuzz dataset .

```

data
  OUTPUTS.ecf_&state._seinunit_fuzz(drop=merge)
  INTERWRK.fuzz_merge_check2(keep=merge)
  ;
merge
INTERWRK.&state._master_seinunit_fuzz(in=a)
INTERWRK.new_seinunit_fuzz_29(in=b)
;
  by sein seinunit;

if a=1 and b=1 then merge=3;
if a=1 and b=0 then merge=1;
if a=0 and b=1 then merge=2;

```

Records only on the original file are passed through untouched

```

if merge=1 then do;
  output OUTPUTS.ecf_&state._seinunit_fuzz;
end;

```

New records are added to the file, including a date stamp.

```

if merge=2 then do;
  DATE_SEINUNIT_FUZZ=date();
  UPDATE_NUMBER_SEINUNIT=&cupdte;
  output OUTPUTS.ecf_&state._seinunit_fuzz;
end;

```

Any matched records indicates an error. This should be improved on: output without abort, then flag QA.

```
output INTERWRK.fuzz_merge_check2;
if merge=3 then do;
  put "%upcase(error) Merge error in the fuzz factors";
  put "%upcase(error) Merge error in the fuzz factors";
  put "%upcase(error) Merge error in the fuzz factors";
  abort abend;
end;
run;
```

Diagnostics and QA on merge. Instructions: check the merge_check file. If no 3s, then OK, otherwise critical error.

```
proc contents data=OUTPUTS.ecf_&state._seinunit_fuzz;
run;

proc freq data=INTERWRK.fuzz_merge_check2;
  tables merge;
run;

proc means data=OUTPUTS.ecf_&state._seinunit_fuzz;
run;

proc freq data=OUTPUTS.ecf_&state._seinunit_fuzz;
  format date_seinunit_fuzz MMDDYYS10.;
  tables UPDATE_NUMBER_SEINUNIT date_seinunit_fuzz;
run;

proc print data=OUTPUTS.ecf_&state._seinunit_fuzz(obs=100);
  format date_seinunit_fuzz MMDDYYS10.;
  id sein seinunit;
run;
```

1.5.2.32 library/sasprogs/01_ecfqa.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* +--< Location: /programs/production/dev1/current/ecf >--- */
/* +--< File: library/sasprogs/01_ecfqa.sas >--- */
/* OLDNAME: leg/1.1.04/library/sasprogs/01_legqa.sas */
/* Time-stamp: <04/05/19 20:43:20 vilhuber> */
/* These mods: <04/04/06 13:51:03 nesto300> */

/* %include 'tempconfig.sas'; */

title1 "QA for State &state.";
```

QA for Module 01: the creation of the ESGAL dataset.

```

title2 "=====LEG Module 01=====";
title3 "Merge of ECF, GAL, by year";
proc print data=INTERWRK.ecfqa_legfreq1
  (rename=(count=within_year_count))
  ;
run;

title3 "Merge of ECF, GAL, across all years";
proc freq data=INTERWRK.ecfqa_legfreq1;
  tables flag_mtch / out=INTERWRK.ecfqa_legfreq1_flagmtch;
run;

/*----- LEG test 1-1 -----*/
/* %register_qa_test(qa_test_name=LEG Test
1-1,qa_module_number=01,status=VERIFY,result_files=interwrk.ecfqa_legfreq1_flagm
tch,notes=Merge of ECF GAL across all years); */

/*----- LEG test 1-2 -----*/

title3 "After merge with GAL";
proc contents data=INTERWRK.esgal;
  run;

proc freq data=INTERWRK.esgal;
  tables es_geoqual / out=INTERWRK.ecfqa_legfreq1_esgal;
run;

/* %register_qa_test(qa_test_name=LEG Test
1-2,qa_module_number=01,status=VERIFY,result_files=interwrk.ecfqa_legfreq1_esgal
,notes=After merge with GAL);*/

```

1.5.2.33 library/sasprogs/02_ecfqa.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/02_ecfqa.sas >--+ */
/* OLDNAME: leg/1.1.04/library/sasprogs/02_legqa.sas */
/* Time-stamp: <04/06/29 10:00:45 vilhuber> */
/* These mods: <04/04/06 13:53:23 nesto300> */

/* %include 'tempconfig.sas'; */

%let module_number=2;

```

QA for Module 02: the creation of the ESGAL dataset.

```

title1 "QA for State &state. ";
title2 "=====LEG Module
&module_number.=====";
title3 "Best of GEO impute";

%let result=%upcase(verify);

data INTERWRK.ecfqa_legprint1;
    set INTERWRK.leg_02(obs=10);
run;

proc print data=INTERWRK.ecfqa_legprint1;
run;

proc freq data=INTERWRK.leg_02;
    tables leg_geoqual / out=INTERWRK.ecfqa_legtest&module_number.1;
    tables es_geoqual / out=INTERWRK.ecfqa_legtest&module_number.2;
    tables year / out=INTERWRK.ecfqa_legtest&module_number.3;
run;

/* This test still needs to be modified !! */

/*----- LEG test 2-1 -----*/

```

How to interpret these tests: [incomplete]

```

%register_qa_test(
    qa_test_name=LEG Test &module_number.-1: Geo quality,
    qa_module_number=0&module_number.,
    status=&result.,
    result_files=INTERWRK.ecfqa_legtest&module_number.1,
    notes=Best of GEO impute);

/*----- LEG test 2-2 -----*/
/* %register_qa_test(qa_test_name=LEG Test
&module_number.-2,qa_module_number=0&module_number.,status=&result.,result_files
=INTERWRK.ecfqa_legtest&module_number.3,notes=Best of GEO impute);*/

/*----- LEG test 2-3 -----*/
/* %register_qa_test(qa_test_name=LEG Test
&module_number.-3,qa_module_number=0&module_number.,status=&result.,result_files
=INTERWRK.ecfqa_legtest&module_number.4,notes=Best of GEO impute);*/

```

1.5.2.34 library/sasprogs/03_ecfqa.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/sasprogs/03_ecfqa.sas >--- */
/* OLDNAME: leg/1.1.04/library/sasprogs/03_legqa.sas */
/* Time-stamp: <04/08/31 10:15:37 vilhuber> */
/* These mods: <04/04/06 17:10:58 nesto300> */

/* %include 'tempconfig.sas'; */

%let module_number=3;

```

QA for Module 03: the creation of the ESGAL dataset.

```

title1 "QA for State &state.";
title2 "=====LEG Module
&module_number.=====";

```

Test 1.

```

proc print data= INTERWRK.ecfqa_legprint2;
    title3 "Creation of impute: where=(es_sic='')";
run;

%let checkobs=0;
%let result=OK;
%let printtable=;

data INTERWRK.ecfqa_legtest&module_number.1
    (keep=es_sic county es_sic_emp);
    set INTERWRK.ecfqa_legprint2 nobs=obs;
    if obs>0 then do;
put obs=;
call symput('checkobs',obs);
call symput('result',"%upcase(verify)");
call symput('printtable',"INTERWRK.ecfqa_legtest&module_number.1");
end;
run;

/*----- LEG test 3-1 -----*/
%register_qa_test(
    qa_test_name=LEG Test &module_number.-1,
    qa_module_number=&module_number.,
    status=&result.,
    result_files=&printtable,
    notes=&checkobs. records with empty es_sic );

```

Test 2: verify only.

```

title3 "Failed records in first COUNTY impute";
proc print data= INTERWRK.ecfqa_legfail1(obs=100);
run;

%let checkobs=0;
%let result=OK;
%let printtable=;

data INTERWRK.ecfqa_legtest&module_number.2;
    set INTERWRK.ecfqa_legfail1 nobs=obs;

    if obs>0 then do;
put obs=;
call symput('checkobs',obs);
call symput('result',"%upcase(verify)");
call symput('printtable',"INTERWRK.ecfqa_legtest&module_number.2");
end;
run;

/*----- LEG test 3-2 -----*/
%register_qa_test(
    qa_test_name=LEG Test &module_number.-2,
    qa_module_number=&module_number.,
    status=&result.,
    result_files=&printtable.,
    notes=Failed records in first county impute);

```

Test 3: verify only.

```
/*----- LEG test 3-3 -----*/

%macro checkds(dsn);

    %if %sysfunc(exist(&dsn)) %then %do;

data _null_;
    nobs=0;
    ds=open("&dsn");
    nobs=attrn(ds,'nobs');
    call symput('checkobs',trim(left(nobs)));
run;

%if &checkobs > 0 %then %do;
    %let result=fail;

    title3 "Failed records in second COUNTY impute for file &dsn";
    proc print data= &dsn;
    run;

/*    %register_qa_test(qa_test_name=LEG Test
&module_number.-3,qa_module_number=&module_number.,status=&result.,
result_files=&dsn,notes=Failed records in second county impute);
*/
    %register_qa_test(qa_test_name=LEG Test
&module_number.-3,qa_module_number=&module_number.,status=&result.,
result_files=,notes=Failed records in second county impute);

    %end;

%end;

    %else %do;

data _null_;
    file print;
    put #3 @10 "Data set &dsn. does not exist";
run;

%end;

    %mend checkds;

%checkds(interwrk.ecfqa_legfail2)
```

Test 4: verify only.

```
***%let result=%upcase(verify);
data INTERWRK.ecfqa_legprint&module_number.;
    set INTERWRK.leg_0&module_number. (obs=10 where=(leg_geoqual in (10, 11)));
run;

proc print data=INTERWRK.ecfqa_legprint&module_number.;
    title3 "LEG obs with imputed county";
run;

/*=====*/
```

Test 5: verify only.

```
data INTERWRK.ecfqa_legprint4;
  set INTERWRK.leg_03(where=(leg_galid="A&stfips.999999999"));
run;

%macro checkds(dsn);

  %if %sysfunc(exist(&dsn)) %then %do;

data _null_;
  nobs=0;
  ds=open("&dsn");
  nobs=attrn(ds,'nobs');
  call symput('checkobs',trim(left(nobs)));
run;

%if &checkobs > 0 %then %do;
  %let result=fail;

  proc print data=INTERWRK.ecfqa_legprint4;
title3 "Errors";
  run;

/*----- LEG test 3-5 -----*/
%register_qa_test(
  qa_test_name=LEG Test &module_number.-4,
  qa_module_number=&module_number.,
  status=&result.,
  result_files=interwrk.ecfqa_legprint4,
  notes=Errors for leg_galid);

%end;
  %end;

%else %do;

data _null_;
  file print;
  put #3 @10 "Data set &dsn. does not exist";
run;

%end;

  %mend checkds;

%checkds(INTERWRK.ecfqa_legprint4)
```

Test 6: verify only.

```

%let result=%upcase(verify);
proc freq data=INTERWRK.leg_0&module_number.;
  title3 "freq of geoqual after impute";
  tables leg_geoqual / out=INTERWRK.ecfqa_legfreq5;
run;

/*----- LEG test 3-6 -----*/
/* %register_qa_test(qa_test_name=LEG Test
&module_number.-6,qa_module_number=&module_number.,status=&result.,result_files=
interwrk.ecfqa_legfreq5,notes=Freq of LEG_GEOQUAL on intermediate file); */

/*----- LEG test 3-7 -----*/
proc sort data=INTERWRK.leg_03 out=leg;
  by sein year quarter seinunit;
  run;

/*      LEG_STRUCTURE eliminated in 3.1.34 */
/* OUTPUTS.ecf_state_leg_structure(keep=sein year quarter seinunit
best_emp1); */

data legtest;
  merge
leg
  INTERWRK.&state._employer_char_unit
  (keep=sein year quarter seinunit best_emp1 )
  ;
  by sein year quarter seinunit;
  leg_county=substr(leg_geo,1,5);
  run;

proc summary data=legtest(where=(leg_geoqual in (10, 11))) nway;
  class leg_county year quarter;
  var best_emp1;
  output out=num sum=impemp;
run;

proc summary data=legtest nway;
  class leg_county year quarter;
  var best_emp1;
  output out=den sum=totemp;
run;

```

Compute the distribution of employment by impute county.

```

%let result=%upcase(pass);
%let notes=Distribution of employment by impute county;
%let warnlevel=.2;
%let printtable=;

data temp(drop= _type_ _freq_);
  merge num den;
  length result $ 10;
  by leg_county year quarter;
  pctimp=impemp/totemp;
  label
pctimp="Percent of employment imputed"
totemp="Total county employment"
;
  format pctimp percent5.2;
  result=upcase('ok');
  if pctimp>&warnlevel. then do;
put "%upcase(warning): "
  "Imputed county employment accounts for " pctimp "% of total
employment in " leg_county year quarter;
result=upcase('warning');
note="Impute may distort employment distribution across counties";
call symput('result',trim(left(result)));
call symput('printtable',"INTERWRK.ecfqa_legtest&module_number.7");
call symput('notes',"Impute may distort employment distribution across
counties");
end;
  run;

proc print data=temp(obs=100);
  run;

data INTERWRK.ecfqa_legtest&module_number.7
  (keep=sein seinunit yyq es_sic leg_county pctimp totemp result);
  set temp(where=(result ne 'OK'));
  length yyq $ 7;
  label yyq="Year:Q";
  yyq=trim(left(year))||"|"||trim(left(quarter));
  run;

%register_qa_test(
  qa_test_name=LEG Test &module_number.-7,
  qa_module_number=&module_number.,
  status=&result.,
  result_files=&printtable.,
  notes=&notes.);

```

Diagnostics for `leg_spike_cutoff` in imputation algorithm: Minimum number of counties in final array of potential counties is 1, ie the largest county. Changing the `leg_spike_cutoff` parameter to a higher value will leave fewer counties in the array and will potentially smooth the final county employment numbers more.

```

/*===== prepare QA =====*/
data trans1;
    set INTERWRK.ecfqa_legprint2(obs=1 keep=p:);
run;

proc transpose data=trans1 out=trans2;
run;

proc sort data=trans2;
where _name_ >='p1' and _name_ <='p99999';
by _name_;
run;

    data _null_;
        set trans2 nobs=totobs;
        call symput('noc',trim(left(totobs)));
    run;

data temp;
    set interwrk.probs_qa;
    array probs(&noc) p1-p&noc;
    largest_prob2=max(of p1-p&noc.);
    do i=1 to &noc;
if probs(i)=. and num=. then num=i-1;
end;
        if num=. then num=&noc;
    run;

proc freq data=temp;
    title "Diagnostics for leg_spike_cutoff in imputation algorithm";
    tables num;
run;

```

1.5.2.35 library/sasprogs/04_ecfqa.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/sasprogs/04_ecfqa.sas >--- */
/* OLDNAME: leg/1.1.04/library/sasprogs/04_legqa.sas */
/* Time-stamp: <04/06/29 14:48:22 vilhuber> */
/* These mods: <04/04/06 17:21:17 nesto300> */

/* %include 'temconfig.sas'; */

%let module_number=4;

```

QA for Module 04: the creation of the ESGAL dataset.

```

title1 "QA for State &state.";
title2 "=====LEG Module
&module_number.=====";

%let result=%upcase(verify);

```

```

/*----- LEG test 4-1 -----*/
proc contents data=OUTPUTS.ecf_&state._leg out=tmp;
run;

data _null_;
  set tmp;
  call symput('legobs',nobs);
run;

%register_qa_test(
  qa_test_name=LEG Test &module_number.-1,
  qa_module_number=0&module_number.,
  status=&result.,
  result_files=,
  notes=Number of observations on LEG dataset is &legobs);

/*----- LEG test 4-2 to 4-4 -----*/

%let result=%upcase(verify);
proc freq data=OUTPUTS.ecf_&state._leg;
  tables es_geo_qual /
out=INTERWRK.ecfqa_legfreq6;
  tables leg_geo_qual /
out=INTERWRK.ecfqa_legfreq7;
  tables leg_flag_geo /
out=INTERWRK.ecfqa_legfreq8;
run;

/*----- LEG test 4-2 -----*/
/* %register_qa_test(qa_test_name=LEG Test
&module_number.-2,qa_module_number=0&module_number.,status=&result.,result_files
=interwrk.ecfqa_legfreq6,notes=Distribution of es_geo_qual on final file); */

/*----- LEG test 4-3 -----*/
/* %register_qa_test(qa_test_name=LEG Test
&module_number.-3,qa_module_number=0&module_number.,status=&result.,result_files
=interwrk.ecfqa_legfreq7,notes=Distribution of leg_geo_qual on final file); */

/*----- LEG test 4-4 -----*/
/* %register_qa_test(qa_test_name=LEG Test
&module_number.-4,qa_module_number=0&module_number.,status=&result.,result_files
=interwrk.ecfqa_legfreq8,notes=Distribution of leg_flag_geo on final file); */

```

```

/*----- LEG test 4-5 -----*/

data ecfqa_legprint6;
    set OUTPUTS.ecf_&state._leg
    (where=(leg_subctygeo=' ' or leg_latitude=.));
run;

%macro checkds(dsn);

    %if %sysfunc(exist(&dsn)) %then %do;

%let checkobs=0;
%let result=OK;
%let note=;

data _null_;
    nobs=0;
    ds=open("&dsn");
    nobs=attrn(ds,'nobs');
    call symput('checkobs',trim(left(nobs)));
run;

%if &checkobs>0 %then %do;
    %if &checkobs > 200 %then %do;
%let result=fail;
%let note=&checkobs observations with missing geography;
    %end;
%else %do;
    %let result=warning;
    %let note=&checkobs missing obs likely due to problem on Version 1
    GAL;
    %end;

    proc print data=&dsn.(obs=10);
title3 "First 10 Observations with missing geography";
    run;

/*----- LEG test 4-1 -----*/
%register_qa_test(qa_test_name=LEG Test &module_number.-5: Missing
Geography,
                qa_module_number=0&module_number.,
                status=&result.,
                result_files=,
                notes=&note.);
%end;

%end;

%else %do;

data _null_;
    file print;
    put #3 @10 "Data set &dsn. does not exist";
run;

%end;

%mend checkds;

%checkds(INTERWRK.ecfqa_legprint6)

```

```

/*----- LEG test 4-6-----*/

data diffcty;
  set OUTPUTS.ecf_&state._leg;
  length es_geocode $11. leg_cty $5. es_cty $5.;
  es_geocode=substr(es_geo,1,11);
  es_cty=substr(es_geocode,1,5);
  leg_cty=substr(leg_geocode,1,5);
  diff=(leg_geocode~=es_geocode);
  diffcty=(leg_cty~=es_cty);
  if es_geocode="" then do;
diffcty=0;
diff=0;
end;
run;

```

Get average differences by quarters.

```

proc means data=diffcty n mean nway;
  title3 "Comparisons of LEG and ES202 geography";
  class year quarter;
  var diffcty diff;
  output out=legdiff mean=;
run;

```

First test: differences in county assignments between LEG and ES202.

```

data INTERWRK.ecfqa_test&module_number._6;
  set legdiff(drop=_type_ _freq_ diff);

run;

%let result=%upcase(pass);
%let notes=ES202 county and LEG county disagree less than 5% in all quarters;

data _null_;
  set INTERWRK.ecfqa_test&module_number._6 end=eof;
  retain cases;
  if diffcty>.05 then do;
put "%upcase(warning): "
  "ES202 county and LEG county disagree " diffcty "% in " year
quarter;
cases+1;
end;
  if eof and cases > 0 then do;
result=upcase('warning');
note="Substantial disagreement between ES202 county and LEG county for
"|| trim(left(cases)) || " cases";
call symput('result',trim(left(result)));
call symput('notes',trim(left(note)));
end;
run;

%register_qa_test(qa_test_name=LEG Test &module_number.-6:Agreement between
ES202 and LEG county,
qa_module_number=0&module_number.,
status=&result.,
result_files=,
notes=&notes);

/*----- LEG test &module_number.-7-----*/

```

Second test: differences in geocoding between LEG and ES202. Ignore in end quarters if GAL time period does not span QWI time period.

```

%let result=%upcase(pass);
%let warnlevel=25;
%let notes=ES202 geography and LEG geography disagree less than &warnlevel.%;
%let printable=;

data ecfqa_legtest&module_number._7pre;
  set legdiff(drop=_type_ _freq_ diffcty);
  format diff percent5.2;
  label diff="(leg_geocode~=es_geocode)";
run;

proc print data=ecfqa_legtest&module_number._7pre;
title3 "differences in geocoding between LEG and ES202.";
run;

data INTERWRK.ecfqa_legtest&module_number._7
  (keep=year quarter diff)
  ;
  set ecfqa_legtest&module_number._7pre;
  if diff>0.&warnlevel. then do;
put %upcase(warning): "
  "ES202 geography and LEG geography disagree " diff "% in " year
quarter;
put "Ignore in end quarters if GAL time period does not span QWI time
period";
result=%upcase('warning');
call symput('result',trim(left(result)));
call symput('notes',"Substantial disagreement between ES202 geography
and LEG geography");
call symput('printtable',"INTERWRK.ecfqa_legtest&module_number._7");
output;
end;
  run;

%register_qa_test(
  qa_test_name=LEG Test &module_number.-7,
  qa_module_number=0&module_number.,
  status=&result.,
  result_files=&printtable.,
  notes=&notes);

```

```

/*----- LEG test 4-8-----*/

proc sort data=OUTPUTS.ecf_&state._leg out=leg;
  by sein seinunit year quarter;
  run;

data leg;
  set leg;
  by sein seinunit year quarter;
  first=(first.seinunit);
  if first.seinunit then nmove=0;
  else nmove=move;
  run;

proc means data=leg n mean std nway;
  title3 "Percentage of firms with different geography than that in previous
quarter";
  class year quarter;
  var nmove;
  output out=INTERWRK.ecfqa_legtest&module_number._8 mean=;
  run;

data INTERWRK.ecfqa_legtest&module_number._8;
  set INTERWRK.ecfqa_legtest&module_number._8(drop=_type_ _freq_);
  format nmove percent5.2;
  label nmove="Percentage of firms with different geography than that in
previous quarter";
  run;

%let result=%upcase(pass);
%let printtable=;
%let warnlevel=15;
%let notes=Quarter to quarter geography changes are less than &warnlevel.%;

data _null_;
  set INTERWRK.ecfqa_legtest&module_number._8;
  if nmove.>.15 then do;
  put "%upcase(warning): "
    "Quarter to quarter geography changes are " nmove "% in " year
quarter;
  result=upcase('warning');
  call symput('result',trim(left(result)));
  call symput('notes',"Large quarter to quarter geography changes");
  call symput('printtable',"INTERWRK.ecfqa_legtest&module_number._8");
  end;
  run;

%register_qa_test(
  qa_test_name=LEG Test &module_number.-8,
  qa_module_number=0&module_number.,
  status=&result,
  result_files=&printtable.,
  notes=&notes);

```

1.5.2.36 library/sasprogs/05_ecfqa.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/sasprogs/05_ecfqa.sas >---+ */
/* Time-stamp: <04/09/02 22:56:10 vilhuber> */
```

```
%let module_number=5;
```

Check on merge of fuzz files.

```
data merge_codes;
  do merge = 1 to 3;
output;
end;
run;

proc freq data=INTERWRK.fuzz_merge_check1;
table merge / out=ecfqa_&module_number.1_fuzz_merge;
run;

%let result=OK;
%let note=;
%let newfuzz=;

%ecfqa_checkfuzz(inoutdata=ecfqa_&module_number.1_fuzz_merge);

%register_qa_test(
  qa_test_name=ECF Test &module_number.-1: Merge for SEIN_FUZZ,
  qa_module_number=&module_number.,
  status=&result.,
  result_files=INTERWRK.ecfqa_&module_number.1_fuzz_merge,
  notes=&note.);
```

Same thing for SEINUNIT fuzz factors.

```
proc freq data=INTERWRK.fuzz_merge_check2;
table merge / out=ecfqa_&module_number.2_fuzz_merge;
run;

%let result=OK;
%let note=;
%let newfuzz=;

%ecfqa_checkfuzz(inoutdata=ecfqa_&module_number.2_fuzz_merge);

%register_qa_test(
  qa_test_name=ECF Test &module_number.-2: Merge for SEINUNIT_FUZZ,
  qa_module_number=&module_number.,
  status=&result.,
  result_files=INTERWRK.ecfqa_&module_number.2_fuzz_merge,
  notes=&note.);
```

Check for usage of LDB data as well. The file ECFQA_LDB_NOT_USED will be non-empty only if no LDB data was used.

```

%let result=OK;
%let note=;

options macrogen symbolgen mlogic;

data _null_;
    set INTERWRK.ecfqa_ldb_not_used;
if _n_=1 and upcase(message)='NO LDB DATA FOUND' then do;
    call symput('result',"WARNING");
    call symput('note',"No LDB was used");
end;
else if ldbobs>=1 and upcase(message) ne 'NO LDB DATA FOUND' then do;
    call symput('result',"OK");
    call symput('note',"LDB was used for "||trim(left(_n_))||" quarters
");
end;

else do;
    call symput('result',"WARNING");
    call symput('note',"Could not determine whether LDB was available");
end;
run;

%let note_srange=;
%let note_erange=;

data _null_;
    set INTERWRK.ecfqa_ldb_not_used end=eof;
if _n_=1 and ldbobs>=1 and upcase(message) ne 'NO LDB DATA FOUND' then do;
    call symput('note_srange',yearqtr);
end;
if eof and upcase(message) ne 'NO LDB DATA FOUND' then do;
    call symput('note_erange',yearqtr);
end;

run;

%put result= &result;
%put note = &note;
%put note_srange = &note_srange.;
%put note_erange = &note_erange;
%let note=&note &note_srange.-&note_erange;

%register_qa_test(
    qa_test_name=ECF Test &module_number.-3: Use of LDB data,
    qa_module_number=&module_number.,
    status=&result.,
    result_files=,
    notes=&note.);

```

1.5.2.37 library/sasprogs/06_ecfqa.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/sasprogs/06_ecfqa.sas >--- */
/* Time-stamp: <04/07/13 09:47:30 vilhuber> */

```

Tabulate quality of SIC and NAICS codes.

```

proc freq data=OUTPUTS.ecf_&state._seinunit;
  table es_naics1997_flag
        es_naics2002_flag
es_naics1997_src
es_naics2002_src
es_naics_aux1997_src
es_naics_aux2002_src
es_naics_ldb1997_src
es_naics_ldb2002_src
es_naics_fnl1997_src;
  table
es_naics_fnl2002_src / out=naics_src;
  table
es_sic_src / out=sic_src
;
run;

```

Print to the PDF the percentage of imputed NAICS and imputed SIC.

1.5.2.38 library/sasprogs/07_ecfqa.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/07_ecfqa.sas >--+ */
/* Time-stamp: <04/11/01 15:52:31 vilhuber> */

%let module_number=7;

```

Check on basic quality measures of ECF.

```

%let result=OK;
%let dataset=INTERWRK.ecfqa_test&module_number.1;

data merge_codes;
  do merge = 1 to 3;
output;
end;
run;

  data &dataset.;
    set INTERWRK.ecf_stacked_01
  (
  where=(duprec=1)
  keep=duprec year quarter sein seinunit
  )
  ;
    run;

data _null_;
  nobs=0;
  ds=open("&dataset");
  nobs=attrn(ds,'nobs');
  if nobs>0 then do;
call symput('checkobs',trim(left(nobs)));
  call symput('result','%upcase(fail)");
  end;
  else do;
call symput('checkobs',"No");
call symput('dataset',"");
  end;
run;

%register_qa_test(
  qa_test_name=ECF Test &module_number.-1: Check for duplicate records,
  qa_module_number=&module_number.,
  status=&result.,
  result_files=&dataset.,
  notes=&checkobs. duplicate records found);

```

1.5.2.39 library/sasprogs/08_ecfqa.sas

```

/* +---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* +---< Location: /programs/production/dev1/current/ecf >---+ */
/* +---< File: library/sasprogs/08_ecfqa.sas >---+ */
/* Time-stamp: <05/03/11 11:05:00 vilhuber> */

%let module_number=8;

```

This module is used only for printing diagnostics. No tests are generated, but a LOT of output is generated for analyst perusal.

Output is only generated if the macro variable diagnostics is set to 'YES' or 'yes' in the parameters file.

```

%let result=OK;

%ecfdiag_01(&diagnostics.);
%ecfdiag_02(&diagnostics.);
%ecfdiag_03(&diagnostics.);
%ecfdiag_04(&diagnostics.);
%ecfdiag_05(&diagnostics.);
%ecfdiag_06(&diagnostics.);
%ecfdiag_07(&diagnostics.);
%ecfdiag_08(&diagnostics.);
%ecfdiag_09(&diagnostics.);
%ecfdiag_10(&diagnostics.);
%ecfdiag_11(&diagnostics.);
%ecfdiag_12(&diagnostics.);
%ecfdiag_13(&diagnostics.);
%ecfdiag_14(&diagnostics.);
%ecfdiag_15(&diagnostics.);
%ecfdiag_16(&diagnostics.);
%ecfdiag_17(&diagnostics.);
%ecfdiag_18(&diagnostics.);
%ecfdiag_19(&diagnostics.);
%ecfdiag_20(&diagnostics.);
%ecfdiag_21(&diagnostics.);
%ecfdiag_22(&diagnostics.);
%ecfdiag_23(&diagnostics.);
%ecfdiag_24(&diagnostics.);
%ecfdiag_25(&diagnostics.);
%ecfdiag_26(&diagnostics.);
%ecfdiag_27(&diagnostics.);
%ecfdiag_28(&diagnostics.);
%ecfdiag_29(&diagnostics.);
%ecfdiag_30(&diagnostics.);
%ecfdiag_31(&diagnostics.);

%register_qa_test(
  qa_test_name=ECF Test &module_number.-1: Diagnostics check,
  qa_module_number=&module_number.,
  status=&result.,
  result_files=,
  notes=Detailed diagnostics were generated: &diagnostics.. To generate
detailed diagnostics, modify parameters file and re-run QA runtime only.);

```

1.5.2.40 library/sasprogs/test.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/sasprogs/test.sas >--+ */
data one;
  do i = 1 to 10;
  output;
  if i = 10 then do;
    call execute('data two;');
    call execute('j=10;');
    call execute('output;');
    call execute('run;');
  end;
end;
run;

proc print data=one;
run;
proc print data=two;
run;

data three;
  call execute('data four;');
  set one;
  call execute('data="one";output;');
  set two;
  call execute('data="two";output;');
  call execute('run;');
run;

data _null_;

  filename="WORK.one";
  dsid=open(filename,'i');
  if (dsid ~= 0) then do;
obs=attrn(dsid,'nobs');
put "FILE " filename "EXISTS with " obs "obs";
end;
  else do;
put "FILE "filename " does NOT exist";
end;
run;
```

1.5.3 SAS macros used

1.5.3.1 library/macros/addnoise.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/addnoise.sas >--+ */
/*-----*/
/* Original Author: Kevin McKinney, Lars Vilhuber */
/* Created : <00/03/31 11:41:52 vilhu001> */
/* Time-stamp: <05/03/11 09:00:22 vilhuber> */
/*-----*/

%put :==== ;
%put :==== Macro: ADDNOISE(outvar,seed,distrib, ;
%put :==== lower,upper[,parm3=c,parm4=d]);
```

This file defines the macro ADDNOISE. It computes from a given random variable draw (provided in SEED) a transformation based on DISTRIBUTION, and leaves it in OUTVAR. Parameters are provided in LOWER, UPPER (required) and PARM3, PARM4 (for certain distributions)

a and b should lie in the following interval

$$[1 < a < b < 2]$$

conceptually $(a-1)$ and $(b-1)$ represent the range in percent distortion allowed

```
/*BEGINCONFIDENTIAL====*/
/* parm3 = beta1 parm4 = beta2 */

%put :==== Available distributions centered around 1;
%put :==== - ramp distribution on [2-b,2-a] and [a,b];
%put :==== - beta distribution, scaled to [2-b,2-a] to [a,b];
%put :==== ;
%put :==== Lower Half of Above Distributions;
%put :==== scaled to integrate to 1;
%put :==== - ramp1 distribution on [2-b,2-a];
%put :==== - beta1 distribution, scaled to [2-b,2-a];
%put :==== ;
%put :==== Upper Half of Above Distributions;
%put :==== scaled to integrate to 1;
%put :==== - ramp2 distribution on [a,b];
%put :==== - beta2 distribution, scaled to [a,b];
%put :==== ;
%put :==== Notes:;
%put :==== - OUTVAR, SEED, DISTRIB are character;
%put :==== - LOWER and UPPER can be numeric constants, or character;
%put :==== - BETA1, BETA2 are numeric;
%put :==== ;
/*ENDCONFIDENTIALEND*/

%macro addnoise(outvar,seed,distrib,lower,upper,parm3,parm4);

%local outvar seed distrib lower upper parm3 parm4;

/*BEGINCONFIDENTIAL====*/
```

RAMP distribution:

```

    %if "&distrib." = "ramp" %then %do;
%put :----- ;
%put :----- Using ramp(&upper,&lower).;
%put :----- PDF: f(&outvar)=
(&upper+&outvar.-2)/((&upper.-&lower.)**2);
%put :----- if &seed. <= .5 ;
%put :----- PDF: f(&outvar)=
(&upper-&outvar.)*2/((&upper.-&lower.)**2);
%put :----- if &seed. > .5 ;
%put :----- ;
    if ( &seed.<=.5) then do;
        &outvar.=(2-&upper.+(2*&seed.*(&upper.-&lower.)**2)**.5);
    end;

    if ( &seed.>.5) then do;
        &outvar=(&upper.-((1-(&seed.-.5)*2)*
            (&upper.-&lower.)**2)**.5);
    end;
%end;

```

Lower RAMP distribution:

```

    %if "&distrib." = "ramp1" %then %do;
%put :----- ;
%put :----- Using ramp1(&upper,&lower).;
%put :----- PDF: f(&outvar)=
2*((&upper+&outvar.-2)*2/((&upper.-&lower.)**2));
%put :----- ;
        &outvar.=(2-&upper.+(&seed.*(&upper.-&lower.)**2)**.5);
%end;

```

Upper RAMP distribution:

```

    %if "&distrib." = "ramp2" %then %do;
%put :----- ;
%put :----- Using ramp(&upper,&lower).;
%put :----- PDF: f(&outvar)=
2*((&upper-&outvar.)*2/((&upper.-&lower.)**2));
%put :----- ;
        &outvar=(&upper.-((1-&seed.)*(&upper.-&lower.)**2)**.5);
%end;

```

Beta distribution:

```

    %if "&distrib." = "beta" %then %do;
%put :----- ;
%put :----- Using beta parameters (&parm3,&parm4).;
%put :----- PDF: f(&outvar.)= (beta( (&outvar. - (2-&upper.)) /
(&upper.-&lower.)))/2;
%put :----- if &seed. <= .5 ;
%put :----- PDF: f(&outvar.)= (beta(&outvar. - &lower.) /
(&upper.-&lower.)))/2;
%put :----- if &seed. > .5 ;
%put :----- ;
    if ( &seed.<=.5) then do;
        &outvar=((&upper.-&lower.)*
            betainv(2*&seed.,&parm3.,&parm4.)+(2-&upper.));
    end;

    if ( &seed.>.5) then do;
        &outvar=(
            (&upper.-&lower.)*
            betainv(2*(&seed.-0.5),&parm4.,&parm3.)+&lower.);
    end;
%end;

```

Lower Beta distribution:

```

    %if "&distrib." = "beta1" %then %do;
    %put :----- ;
    %put :----- Using beta parameters (&parm3,&parm4).;
    %put :----- PDF: f(&outvar.)= beta( (&outvar. - (2-&upper.)) /
    (&upper.-&lower.));
    %put :----- ;
    %outvar=((&upper.-&lower.)*betainv(&seed.,&parm3.,&parm4.)+(2-&upper.));
    %end;

```

Upper Beta distribution:

```

    %if "&distrib." = "beta2" %then %do;
    %put :----- ;
    %put :----- Using beta parameters (&parm3,&parm4).;
    %put :----- PDF: f(&outvar.)= beta(&outvar. - &lower.) /
    (&upper.-&lower.));
    %put :----- ;
    %outvar=((&upper.-&lower.)*betainv(&seed.,&parm4.,&parm3.)+&lower.);
    %end;

    %*                               ;
    %* -----end of distributions----- ;
    %* -----outputting result ----- ;
    %*                               ;
    /*====CONFIDENTIALEND*/

    %mend;

```

1.5.3.2 library/macros/bldstr.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/bldstr.sas >--- */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/*
*****
*** MACRO to create a long series of strings ****
*****/
/*
*** if EIN then set einvar to yes ***/
%macro bldstr(prefix,einvar);
%do i=&start_year %to &end_year;
&prefix.&i.1 &prefix.&i.2 &prefix.&i.3 &prefix.&i.4
%end;
%mend;

```

1.5.3.3 library/macros/check_ein.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/check_ein.sas >--+ */
/* Time-stamp: <04/03/02 15:31:58 vilhuber> */
/*-----
 * EIN Checking algorithm used in ECF read in
 * Variables Used
 * temp_char
 * ein_bad
 * ein_num
 * ein_char2
 * valid_ein
 * ein_defect
 *-----*/

%macro check_ein(ein);

    *drop rk1 ein_num valid_ein ein_bad;
```

Check for Bad EIN data

```
ein_bad=0;
do i=1 to 9;
    rk1=rank(substr(&ein,i,1));
    if (rk1 < 48 or rk1 > 57) then ein_bad=1;
end;
```

Store the original EIN for cleanup.

```
temp_ein=&ein;
ein_num=.;
if ein_bad=0 then ein_num=input(&ein,9.);
ein_char2=substr(&ein,1,2);
```

Clean up the EIN

```
/* Updated Sept 2002 */  
/* Prefix list source current as of Jan 2002 */
```

```
valid_ein=0;  
if ein_char2="01" then valid_ein=1;  
else if ein_char2="02" then valid_ein=1;  
else if ein_char2="03" then valid_ein=1;  
else if ein_char2="04" then valid_ein=1;  
else if ein_char2="05" then valid_ein=1;  
else if ein_char2="06" then valid_ein=1;  
else if ein_char2="10" then valid_ein=1;  
else if ein_char2="11" then valid_ein=1;  
else if ein_char2="12" then valid_ein=1;  
else if ein_char2="13" then valid_ein=1;  
else if ein_char2="14" then valid_ein=1;  
else if ein_char2="15" then valid_ein=1;  
else if ein_char2="16" then valid_ein=1;  
else if ein_char2="20" then valid_ein=1;  
else if ein_char2="21" then valid_ein=1;  
else if ein_char2="22" then valid_ein=1;  
else if ein_char2="23" then valid_ein=1;  
else if ein_char2="24" then valid_ein=1;  
else if ein_char2="25" then valid_ein=1;  
else if ein_char2="26" then valid_ein=1;  
else if ein_char2="27" then valid_ein=1;  
else if ein_char2="30" then valid_ein=1;  
else if ein_char2="31" then valid_ein=1;  
else if ein_char2="32" then valid_ein=1;  
else if ein_char2="33" then valid_ein=1;  
else if ein_char2="34" then valid_ein=1;  
else if ein_char2="35" then valid_ein=1;  
else if ein_char2="36" then valid_ein=1;  
else if ein_char2="37" then valid_ein=1;  
else if ein_char2="38" then valid_ein=1;  
else if ein_char2="39" then valid_ein=1;  
else if ein_char2="40" then valid_ein=1;  
else if ein_char2="41" then valid_ein=1;  
else if ein_char2="42" then valid_ein=1;  
else if ein_char2="43" then valid_ein=1;  
else if ein_char2="44" then valid_ein=1;  
else if ein_char2="45" then valid_ein=1;  
else if ein_char2="46" then valid_ein=1;  
else if ein_char2="47" then valid_ein=1;  
else if ein_char2="48" then valid_ein=1;
```

```

else if ein_char2="50" then valid_ein=1;
else if ein_char2="51" then valid_ein=1;
else if ein_char2="52" then valid_ein=1;
else if ein_char2="53" then valid_ein=1;
else if ein_char2="54" then valid_ein=1;
else if ein_char2="55" then valid_ein=1;
else if ein_char2="56" then valid_ein=1;
else if ein_char2="57" then valid_ein=1;
else if ein_char2="58" then valid_ein=1;
else if ein_char2="59" then valid_ein=1;
else if ein_char2="60" then valid_ein=1;
else if ein_char2="61" then valid_ein=1;
else if ein_char2="62" then valid_ein=1;
else if ein_char2="63" then valid_ein=1;
else if ein_char2="64" then valid_ein=1;
else if ein_char2="65" then valid_ein=1;
else if ein_char2="66" then valid_ein=1;
else if ein_char2="67" then valid_ein=1;
else if ein_char2="68" then valid_ein=1;
else if ein_char2="69" then valid_ein=1;
else if ein_char2="70" then valid_ein=1;
else if ein_char2="71" then valid_ein=1;
else if ein_char2="72" then valid_ein=1;
else if ein_char2="73" then valid_ein=1;
else if ein_char2="74" then valid_ein=1;
else if ein_char2="75" then valid_ein=1;
else if ein_char2="76" then valid_ein=1;
else if ein_char2="77" then valid_ein=1;
else if ein_char2="80" then valid_ein=1;
else if ein_char2="81" then valid_ein=1;
else if ein_char2="82" then valid_ein=1;
else if ein_char2="83" then valid_ein=1;
else if ein_char2="84" then valid_ein=1;
else if ein_char2="85" then valid_ein=1;
else if ein_char2="86" then valid_ein=1;
else if ein_char2="87" then valid_ein=1;
else if ein_char2="88" then valid_ein=1;
else if ein_char2="90" then valid_ein=1;
else if ein_char2="91" then valid_ein=1;
else if ein_char2="92" then valid_ein=1;
else if ein_char2="93" then valid_ein=1;
else if ein_char2="94" then valid_ein=1;
else if ein_char2="95" then valid_ein=1;
else if ein_char2="96" then valid_ein=1;
else if ein_char2="97" then valid_ein=1;
else if ein_char2="98" then valid_ein=1;
else if ein_char2="99" then valid_ein=1;

```

Basic EIN clean-up

```

ein_defect=0;
if temp_ein='999999999' or temp_ein='000000000' then ein_defect=1;
else if ein_bad=1 then ein_defect=2;
else if ein_num<10000000 then ein_defect=3;
else if valid_ein=0 then ein_defect=4;
%mend;

```

1.5.3.4 library/macros/choose_gal2.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/choose_gal2.sas >--+ */
%macro choose_gal2;
```

Code for GAL 1 (obsolete).

```
    %if (&gal_version=1) %then %do;

data gal;
    set INPUTS.gal_&state
(keep=a_scccc a_&subcty a_latitude a_longitude galid a_wib
a_msapmsa
a_geocode a_block a_block_src
rename=(a_latitude=leg_latitude a_longitude=leg_longitude
a_wib=leg_wib
a_msapmsa=leg_msapmsa a_geocode=leg_geocode a_block=leg_block
a_block_src=leg_block_src)
);
run;

    %end;
```

Code for GAL 2 (transitional)

```
    %else %if (&gal_version=2) %then %do;

    data gal(drop=a_geocode a_block);
        set INPUTS.gal_&state
(keep=a_&subcty a_latitude a_longitude galid a_wib a_msapmsa
a_geocode a_block a_block_src
rename=(a_latitude=leg_latitude a_longitude=leg_longitude
a_wib=leg_wib
a_msapmsa=leg_msapmsa a_geocode=leg_geocode a_block=leg_block
a_block_src=leg_block_src)
)
        INPUTS.gal_&state._tccb
(keep=a_&subcty a_latitude a_longitude galid a_wib a_msapmsa
a_geocode a_block a_block_src
rename=(a_latitude=leg_latitude a_longitude=leg_longitude
a_wib=leg_wib
a_msapmsa=leg_msapmsa a_geocode=leg_geocode a_block=leg_block
a_block_src=leg_block_src)
);
        length a_scccc $5.;
        a_scccc=substr(leg_geocode,1,5);
        run;

        proc sort data=gal;
        by galid;
        run;

    %end;
```

Code for GAL 3.

```

%else %if (&gal_version=3) %then %do;

    data gal(drop=a_geocode a_block);
    set INPUTS.gal_&state._&gal_geocode_year
(keep=a_&subcty a_latitude a_longitude galid a_wib a_msapmsa
    a_geocode a_block a_block_src
    /* variables added per Marc Roemer, 2005-01-26 */
    a_block_suf1 a_block_suf2
rename=(a_latitude=leg_latitude a_longitude=leg_longitude
a_wib=leg_wib
    a_msapmsa=leg_msapmsa a_geocode=leg_geocode
a_block=leg_block
    a_block_src=leg_block_src
    /* variables added per Marc Roemer, 2005-01-26 */
    a_block_suf1=leg_block_suf1
    a_block_suf2=leg_block_suf2
    )
)
    INPUTS.gal_&state._&gal_geocode_year._tccb
(keep=a_&subcty a_latitude a_longitude galid a_wib a_msapmsa
a_geocode a_block a_block_src
rename=(a_latitude=leg_latitude a_longitude=leg_longitude
a_wib=leg_wib
a_msapmsa=leg_msapmsa a_geocode=leg_geocode a_block=leg_block
a_block_src=leg_block_src)
);
    length a_scccc $5.;
    a_scccc=substr(leg_geocode,1,5);
    run;

    proc sort data=gal;
    by galid;
    run;

    %end;

%mend;

```

1.5.3.5 library/macros/choose_gal.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/choose_gal.sas >---+ */
%macro choose_gal;

    %if (&gal_version=1) %then %do;

data gal;
    set INPUTS.gal_&state(keep=galid a_geoqual a_geocode a_block);
    length a_geo $15.;
    a_geo=a_geocode||a_block;
    run;

    proc sort data=gal;
    by galid;
    run;

    %end;

    %else %if (&gal_version=2) %then %do;

    data gal(drop=a_geocode a_block);
    set INPUTS.gal_&state(keep=galid a_geoqual a_geocode a_block)
        INPUTS.gal_&state._tccb(keep=galid a_geoqual a_geocode a_block);
    length a_geo $15;
    a_geo=a_geocode||a_block;
    run;

    proc sort data=gal;
    by galid;
    run;

    %end;

    %else %if (&gal_version=3) %then %do;

    data gal(drop=a_geocode a_block);
    set INPUTS.gal_&state._&gal_geocode_year(keep=galid a_geoqual
a_geocode a_block)
        INPUTS.gal_&state._&gal_geocode_year._tccb(keep=galid a_geoqual
a_geocode a_block);
    length a_geo $15;
    a_geo=a_geocode||a_block;
    run;

    proc sort data=gal;
    by galid;
    run;

    %end;

%mend;
```

1.5.3.6 library/macros/create_esgal.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/create_esgal.sas >---+ */
/* Time-stamp: <04/10/28 18:19:01 vilhuber> */
%macro create_esgal;
```

First process years in which there is es202 data but no address information and therefore no GAL xwalk. (eg NC, MN) Keep ECF, separately by year to define universe of sein seinunit year quarter obs.

```
    %if &start_year < &gal_start_year %then %do;
%let a=&start_year.;
%let b=%eval(&gal_start_year-1);

%do year=&a %to &b.;
    %create_pseudo_esgal(year=&year.);

/*===== preparing QA =====*/
proc freq data=esgal&year;
    title "ECF only=1, ECF and xwlk=2 for &year";
    tables flag_mtch / out=tmp;
run;
data tmp;
    set tmp;
    year=&year.;
run;
proc append base=WORK.ecfqa_legfreq1 data=tmp;
run;
/*===== end QA =====*/

proc append base=esgal data=esgal&year;
run;
%end; /* end year loop */
%end; /* end of loop without XWALK, pre */
```

Merge GAL xwalk (keeping galid), and ECF, separately by year, to define universe of sein seinunit year quarter obs.

```

%do year=&gal_start_year %to &gal_end_year;

/* proc sort data=INPUTS.xwalk_e&year out=xwlc_e&year; */
proc sort data=INPUTS.gal_&state._&gal_geocode_year._xwalk_&year
out=xwlc_e&year;

    by sein year quarter seinunit;
    run;

data esgal&year
    (keep=sein year quarter seinunit es_sic sic es_county galid
    e_flag flag_mtch);

    merge
xwlc_e&year
(keep=galid sein seinunit year quarter e_flag
in=in1)
/* replaced 3.1.32 */
/* INTERWRK.ecf_&state._leg_structure */
INTERWRK.&state._employer_char_unit
(keep=sein seinunit year quarter county sic
where=(year=&year) rename=(county=es_county)
in=in2)
;

    by sein year quarter seinunit;

    es_county=put(es_county, $cty_tmp.);
    if es_county="ZZZ" then es_county="";

    es_sic=substr(sic,1,2);
    if es_sic in ('00' '98' '99') then es_sic='';

    if in2 then flag_mtch=1;
    if in1 and in2 then flag_mtch=2;

    if in2;
    run;

/*===== preparing QA =====*/
proc freq data=esgal&year;
    title "ECF only=1, ECF and xwlc=2 for &year";
    tables flag_mtch / out=tmp;
    run;
data tmp;
    set tmp;
    year=&year.;
    run;
proc append base=work.ecfqa_legfreq1 data=tmp;
    run;
/*===== end QA =====*/

    proc append base=esgal data=esgal&year;
    run;

%end; /* end of within-GAL year loop */

```

This function used to be run only if the ES202 started before the GAL. To allow for asynchronous production of the GAL, Lars modified this to run also for other years outside the GAL range.

```

        %if &end_year > &gal_end_year %then %do;
%let a=%eval(&gal_end_year+1);
%let b=&end_year.;

%do year=&a %to &b.;
    %create_pseudo_esgal(year=&year.);

/*===== preparing QA =====*/
proc freq data=esgal&year;
    title "ECF only=1, ECF & xwlk=2 for &year";
    tables flag_mtch / out=tmp;
run;
data tmp;
    set tmp;
    year=&year.;
run;
proc append base=work.ecfqa_legfreq1 data=tmp;
run;
/*===== end QA =====*/

proc append base=esgal data=esgal&year;
run;

%end; /* end year loop */
%end; /* end of loop without XWALK, post */

/*===== preparing QA =====*/

data INTERWRK.ecfqa_legfreq1;
    set WORK.ecfqa_legfreq1;
run;

%mend;

```

1.5.3.7 library/macros/create_pseudo_esgal.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/create_pseudo_esgal.sas >--- */
/* Time-stamp: <04/07/13 13:25:44 vilhuber> */

```

This macro creates a file that looks like a ECF-GAL merge for years that do not have a GAL XWALK.

```

%macro create_pseudo_esgal(year=);

data esgal&year
    (keep=sein year quarter seinunit es_sic sic
     es_county galid e_flag flag_mtch);
    set
        /* replaced 3.1.32 */
/* INTERWRK.ecf_&state._leg_structure */
INTERWRK.&state._employer_char_unit
(keep=sein seinunit year quarter county sic
 where=(year=&year) rename=(county=es_county) in=in2);

    by sein year quarter seinunit;

```

Define place holders for variables that exist on leg xwlk in later years.

```
length galid $15. e_flag $1.;
galid='';
e_flag='';
**** in1=1 if observation is found on leg xwlk ****;
in1=0;

es_county=put(es_county, $cty_tmp.);
if es_county="ZZZ" then es_county="";

es_sic=substr(sic,1,2);
if es_sic in ('00' '98' '99') then es_sic='';

if in1 then flag_mtch=1;
if in2 then flag_mtch=2;
if in1 and in2 then flag_mtch=3;
****if not in2 and in3 then flag_mtch=4;
****if in3;

if not in2 and in1 then flag_mtch=4;
if in2;

run;
%mend;
```

1.5.3.8 library/macros/ecfdiag_01.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_01.sas >--+ */
/*----- LEGACY NAME: 01_read_es202.sas -----*/
```

This macro contains the diagnostic output formerly contained in 01_read_es202.sas It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_01(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_01 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 01";

proc contents data=INTERWRK.ecf_stacked_01;
run;

proc means data=INTERWRK.ecf_stacked_01;
run;

proc freq data=INTERWRK.ecf_stacked_01;
tables year*quarter duprec sein_bad seinunit_bad /v5fmt;
tables owner_code empl_month1_flg empl_month2_flg empl_month3_flg
total_wages_flg multi_unit_code auxiliary_code /v5fmt;
tables county county_temp seinunit /v5fmt;
tables sic naics naics_aux ein_char2 ein_bad valid_ein ein_bad*valid_ein
ein_defect /missing v5fmt;
run;

proc univariate data=INTERWRK.ecf_stacked_01;
var empl_month1 empl_month2 empl_month3 total_wages;
run;

proc print data=INTERWRK.ecf_stacked_01(obs=1000);
title3 "The first 1000 observations";
var sein year quarter seinunit owner_code sic
NAICS NAICS_AUX county total_wages empl_month1 empl_month2
empl_month3;
run;

```

Restrict Sample size for Printouts .

```

options obs=&listsize;

proc print data=INTERWRK.ecf_stacked_01(where=(sein_bad=1));
  title3 "Observations where the SEIN is bad (digits not between 0-9)";
  var sein year quarter seinunit owner_code sic county total_wages
empl_month1 empl_month2 empl_month3;
run;
proc print data=INTERWRK.ecf_stacked_01(where=(seinunit_bad=1));
  title3 "Observations where the SEINUNIT is bad (digits not between 0-9)";
  var sein year quarter seinunit owner_code sic county total_wages
empl_month1 empl_month2 empl_month3;
run;
proc print data=INTERWRK.ecf_stacked_01(where=(duprec=1));
  title3 "SEIN YEAR QUARTER SEINUNIT records that have duplicates";
  var sein year quarter seinunit owner_code sic county total_wages
empl_month1 empl_month2 empl_month3;
run;

proc print data=INTERWRK.ecf_stacked_01(where=(ein_defect=1) );
  title3 "Observations where ein_defect=1";
  var sein year quarter seinunit ein_defect valid_ein ein_bad ein temp_ein;
run;
proc print data=INTERWRK.ecf_stacked_01(where=(ein_defect=2) );
  title3 "Observations where ein_defect=2";
  var sein year quarter seinunit ein_defect valid_ein ein_bad ein temp_ein;
run;
proc print data=INTERWRK.ecf_stacked_01(where=(ein_defect=3) );
  title3 "Observations where ein_defect=3";
  var sein year quarter seinunit ein_defect valid_ein ein_bad ein temp_ein;
run;
proc print data=INTERWRK.ecf_stacked_01(where=(ein_defect=4) );
  title3 "Observations where ein_defect=4";
  var sein year quarter seinunit ein_defect valid_ein ein_bad ein temp_ein;
run;

options obs=max;

```

END QA PREP

```

%end;
%mend;

```

1.5.3.9 library/macros/ecfdiag_02.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_02.sas >--+ */
/*----- LEGACY NAME: 02_naics_clean.sas -----*/

%macro ecfdiag_02(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_02 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 02";

proc contents data=INTERWRK.naics_clean_02;
run;

proc freq data=INTERWRK.naics_clean_02;
tables merge;
run;

proc print data=INTERWRK.naics_clean_02(obs=1000);
run;

%end;
%mend;
```

1.5.3.10 library/macros/ecfdiag_03.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_03.sas >--+ */
/*----- LEGACY NAME: 03_naics_aux_clean.sas -----*/
```

This macro contains the diagnostic output formerly contained in 03_naics_aux_clean.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```
%macro ecfdiag_03(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_03 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 03";
```

Final output data set

```

proc contents data=INTERWRK.naics_aux_clean_03;
run;

proc freq data=INTERWRK.naics_aux_clean_03;
  tables merge;
run;

proc print data=data=INTERWRK.naics_aux_clean_03(obs=1000);
run;

%end;
%mend;

```

1.5.3.11 library/macros/ecfdiag_04.sas

```

/* +---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* +---< Location: /programs/production/dev1/current/ecf >---+ */
/* +---< File: library/macros/ecfdiag_04.sas >---+ */
/*----- LEGACY NAME: 04_sic_clean.sas -----*/

```

This macro contains the diagnostic output formerly contained in 04_sic_clean.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_04(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_04 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 04";

/** DIAGNOSTIC OUTPUT **/

proc contents data=INTERWRK.temp5_04;
run;

proc freq data=INTERWRK.temp5_04;
  tables merge candidate sic_invalid*sic_valid;
run;

proc freq data=INTERWRK.temp5_04(where=(sic_invalid~="0"));
  tables sic_clean*sic_invalid;
run;

proc print data=INTERWRK.temp5_04(where=(candidate=1));
  var sein seinunit sic_invalid sic_valid sic sic_clean sic_impute;
run;

```

Final output data set

```

proc contents data=INTERWRK.sic_clean_04;
run;

proc freq data=INTERWRK.sic_clean_04;
  tables merge;
run;

proc print data=INTERWRK.sic_clean_04(obs=1000);
run;

proc sort data=temp6 out=INTERWRK.sic_clean_04;
  by sein year quarter seinunit;
run;

%end;
%mend;

```

1.5.3.12 library/macros/ecfdiag_05.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_05.sas >--+ */
/*----- LEGACY NAME: 05_merge_ind.sas -----*/

```

This macro contains the diagnostic output formerly contained in 05_merge_ind.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_05(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_05 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 05";

```

Diagnostics

```

proc contents data=INTERWRK.ecf_stacked_clean_05;
run;

proc freq data=INTERWRK.ecf_stacked_clean_05;
  tables merge4;
  tables sic_equal naics1997_equal naics2002_equal naics_aux1997_equal
naics_aux2002_equal;
  tables naics_1997_invalid naics_2002_invalid;
  tables naics_aux_1997_invalid naics_aux_2002_invalid;
  tables sic_invalid;
run;

/* when do we receive each industry variable */

proc freq data=INTERWRK.ecf_stacked_clean_05;
  title3 "Industry codes available by year";
  tables year*naics_1997_invalid;
  tables year*naics_2002_invalid;
  tables year*naics_aux_1997_invalid;
  tables year*naics_aux_2002_invalid;
  tables year*sic_invalid;
  tables naics1997_clean naics2002_clean naics_aux1997_clean
naics_aux2002_clean sic_clean;
run;

proc print data=INTERWRK.ecf_stacked_clean_05(obs=1000);
  id sein year quarter seinunit;
  var sic sic_clean naics naics1997_clean naics2002_clean naics_aux
naics_aux1997_clean naics_aux2002_clean;
run;

%end;
%mend;

```

1.5.3.13 library/macros/ecfdiag_06.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_06.sas >--+ */
/*----- LEGACY NAME: 06_num_records.sas -----*/

```

This macro contains the diagnostic output formerly contained in 06_num_records.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_06(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_06 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 06";

```

Diagnostics.

```

proc contents INTERWRK.count_data_06;
run;

proc freq;
  title3 "SEIN YEAR QUARTER SEINUNIT records";
  tables num_records all_miss_pay all_miss_emp1 all_miss_emp2 all_miss_emp3
         all_miss_sic all_miss_county all_miss_owner_code /v5fmt;
  tables all_miss_naics1997 all_miss_naics2002 all_miss_naics_aux1997
         all_miss_naics_aux2002 /v5fmt;
run;

proc print data=INTERWRK.count_data_06(obs=500);
id sein year quarter seinunit;
run;

%end;
%mend;

```

1.5.3.14 library/macros/ecfdiag_07.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_07.sas >--+ */
/*----- LEGACY NAME: 07_rm_master.sas -----*/

```

This macro contains the diagnostic output formerly contained in 07_rm_master.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_07(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_07 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 07";

```

Diagnostics.

```

proc contents data=INTERWRK.no_master_07;
run;

proc freq data=INTERWRK.no_master_07;
  tables year*no_wages year*no_emp1 year*no_emp2 year*no_emp3 year*no_sic
  year*no_county year*no_owner_code /v5fmt;
  tables year*no_naics1997 year*no_naics2002 year*no_naics_aux1997
  year*no_naics_aux2002 /v5fmt;
  tables multi_unit*no_wages multi_unit*no_emp1 multi_unit*no_emp2
  multi_unit*no_emp3 multi_unit*no_sic multi_unit*no_county
  multi_unit*no_owner_code /v5fmt;
  tables multi_unit*no_naics1997 multi_unit*no_naics2002
  multi_unit*no_naics_aux1997 multi_unit*no_naics_aux2002 /v5fmt;
  tables impute_wage impute_emp1 impute_emp2 impute_emp3 impute_sic
  impute_county impute_owner_code /v5fmt;
  tables impute_naics1997 impute_naics2002 impute_naics_aux1997
  impute_naics_aux2002 /v5fmt;
  tables impute_wage*no_wages impute_emp1*no_emp1 impute_emp2*no_emp2
  impute_emp3*no_emp3 impute_sic*no_sic impute_county*no_county
  impute_owner_code*no_owner_code /v5fmt;
  tables impute_naics1997*no_naics1997 impute_naics2002*no_naics2002
  impute_naics_aux1997*no_naics_aux1997 impute_naics_aux2002*no_naics_aux2002
  /v5fmt;
  tables seinunit_type /v5fmt;
  tables num_records*seinunit_type /missing v5fmt;
  tables num_estabs*seinunit_type /missing v5fmt;
  tables multi_unit year*multi_unit /v5fmt;
  tables num_estabs /v5fmt;
  tables sic naics1997 naics2002 naics_aux1997 naics_aux2002 county owner_code
  /v5fmt;
  tables master_empl_month1_flg*multi_unit master_empl_month2_flg*multi_unit
  master_empl_month3_flg*multi_unit
  master_total_wages_flg*multi_unit master_multi_unit_code*multi_unit /
  missing v5fmt;
run;

proc print data=INTERWRK.no_master_07(obs=500);
id sein year quarter seinunit;
run;
proc print data=INTERWRK.no_master_07(where=(impute_wage=1));
id sein year quarter seinunit;
run;

%end;
%mend;

```

1.5.3.15 library/macros/ecfdiag_08.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/ecfdiag_08.sas >--- */
/*----- LEGACY NAME: 08_sein_totals.sas -----*/

```

This macro contains the diagnostic output formerly contained in 08_sein_totals.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_08(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_08 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 08";

```

Diagnostics.

```

proc contents data=INTERWRK.sein_totals_08;
title3 "ES202 Data";
run;
proc contents data=INTERWRK.sein_list_202_08;
run;

proc print data=INTERWRK.sein_totals_08(obs=600);
title3 "ES202 SEIN YEAR QUARTER data";
id sein year quarter;
run;
proc print data=INTERWRK.sein_list_202_08(obs=100);
title3 "ES202 SEIN data";
id sein;
run;

proc freq data=INTERWRK.sein_list_202_08;
title3 "ES202 SEIN data";
tables ever_multi*ever_wages ever_multi*ever_emp1 ever_multi*ever_emp2
ever_multi*ever_emp3;
tables multi_first_year*multi_first_quarter;
run;

proc univariate data=INTERWRK.sein_totals_08;
title "ES202 SEIN YEAR QUARTER data";
var sein_emp1 sein_emp2 sein_emp3 sein_wages;
run;

proc means data=INTERWRK.sein_list_202_08;
var ever_multi ever_wages ever_emp1 ever_emp2 ever_emp3;
run;

```

START QA PREP

```

title3 "data=INTERWRK.seinunit_202_UI_08";

proc contents data=INTERWRK.seinunit_202_UI_08;
run;

proc freq data=INTERWRK.seinunit_202_UI_08;
  title3 "Merge of 202 SEIN YEAR QUARTER SEINUNIT data with SEIN YEAR QUARTER
  UI/202 data";
  tables _merge ever_202 ever_UI ever_202*in_UI*in_202 /v5fmt;
  tables yr_qtr*source /v5fmt;
  tables yr_qtr*in_UI /v5fmt;
  tables yr_qtr*in_202 /v5fmt;
  tables yr_qtr*ever_UI /v5fmt;
  tables yr_qtr*ever_202 /v5fmt;
  tables source*ever_multi /v5fmt;
  tables source*ever_wages /v5fmt;
  tables source*ever_emp1 /v5fmt;
  tables source*ever_emp2 /v5fmt;
  tables source*ever_emp3 /v5fmt;
  tables multi_unit*ever_multi /v5fmt;
  tables in_UI*ever_UI in_202*ever_202 /v5fmt;
run;

proc means data=INTERWRK.seinunit_202_UI_08;
  var empl_month1 empl_month2 empl_month3 total_wages;
  var sein_emp1 sein_emp2 sein_emp3 sein_wages;
  var seinsize_m seinsize_e seinsize_b payroll1;
run;

proc print data=INTERWRK.seinunit_202_UI_08(obs=100);
  id sein year quarter seinunit;
run;

```

END QA PREP

```

%end;
%mend;

```

1.5.3.16 library/macros/ecfdiag_09.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_09.sas >--+ */
/*----- LEGACY NAME: 09_best_vars.sas -----*/

```

This macro contains the diagnostic output formerly contained in 09_best_vars.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_09(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_09 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 09";

```

Diagnostics.

```

proc contents data=INTERWRK.best_vars_09;
run;

proc freq data=INTERWRK.best_vars_09;
  tables in_UI in_202 info_202*in_202 info_202*in_ui in_202*in_ui best_flag
  info_202*in_ui*best_flag;
run;

proc means data=INTERWRK.best_vars_09;
  var best_wages best_emp1 best_emp2 best_emp3;
run;

/* Restrict sample size for data listings */
options obs=&listsize;

proc print data=INTERWRK.best_vars_09(where=(best_flag=1));
  title3 "Section 1";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=2));
  title3 "Section 2";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=3));
  title3 "Section 3";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=4));
  title3 "Section 4";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=5));
  title3 "Section 5";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=6));
  title3 "Section 6";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

```

```

proc print data=INTERWRK.best_vars_09(where=(best_flag=7));
  title3 "Section 7";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=8));
  title3 "Section 8";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=9));
  title3 "Section 9";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=10));
  title3 "Section 10";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=11));
  title3 "Section 11";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.best_vars_09(where=(best_flag=0));
  title3 "Missing Data";
  id sein year quarter seinunit;
  var sein_wages info_202 in_UI sein_emp1 sein_emp2 sein_emp3 payroll
  seinsize_m seinsize_b seinsize_e total_wages empl_month1 empl_month2
  empl_month3 best_wages best_emp1 best_emp2 best_emp3;
run;

options obs=max;

%end;
%mend;

```

1.5.3.17 library/macros/ecfdiag_10.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/ecfdiag_10.sas >--- */
/*----- LEGACY NAME: 10_select_records.sas -----*/

```

This macro contains the diagnostic output formerly contained in 10_select_records.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_10(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_10 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 10";

```

Get some statistics on which firms have data in the 202 that can be used . to determine firm structure .

```

proc contents data=INTERWRK.alldata_10;
run;

proc contents data=INTERWRK.special_handle_list_10;
run;

proc freq data=INTERWRK.alldata_10;
tables impute_data no_data no_get_data qtime_master qtime_first;
tables special_handle special_handle*no_get_data;
tables data_avail;
tables best_flag*special_handle;
run;

proc freq data=INTERWRK.special_handle_list_10;
tables special_handle qtime_master;
tables qtime_master*special_handle;
run;

proc print data=INTERWRK.special_handle_list_10(obs=1000);
title3 "SPECIAL_HANDLE_LIST_10 data listing (first 1000 obs)";
run;

proc contents INTERWRK.special_handle_history_10;
run;

proc print data=INTERWRK.special_handle_history_10();
title3 "SPECIAL_HANDLE_HISTORY_10 data listing";
id sein year quarter seinunit;
run;

%end;
%mend;

```

1.5.3.18 library/macros/ecfdiag_11.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_11.sas >--+ */
/*----- LEGACY NAME: 11_special_handle.sas -----*/

```

This macro contains the diagnostic output formerly contained in 11_special_handle.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_11(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_11 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 11";

```

Diagnostics.

```

proc contents data=INTERWRK.special_handle_11();
run;

proc freq data=INTERWRK.special_handle_11();
tables stop i;
run;

proc print data=INTERWRK.special_handle_11();
title3 " All Records on Special Handle";
var sein year quarter seinunit special_handle i qtime_master year_found
quarter_found stop total_wages empl_month1 empl_month2 empl_month3
wages_202 emp1_202 emp2_202 emp3_202 wages_UI emp1_UI emp2_UI emp3_UI;
run;

proc print data=INTERWRK.special_handle_11(where=(stop=0));
title3 "Records Where stop=0";
var sein year quarter seinunit special_handle i qtime_master year_found
quarter_found stop total_wages empl_month1 empl_month2 empl_month3
wages_202 emp1_202 emp2_202 emp3_202 wages_UI emp1_UI emp2_UI emp3_UI;
run;

```

Check if all records are in output dataset. KEVIN, what are you actually checking here?

```

data check1;
set INTERWRK.special_handle_11;
by sein year quarter;

if last.quarter then output;
run;

%end;
%mend;

```

1.5.3.19 library/macros/ecfdiag_12.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_12.sas >--+ */
/*----- LEGACY NAME: 12_distribute.sas -----*/

```

This macro contains the diagnostic output formerly contained in 12_distribute.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_12(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_12 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 12";

proc print data=INTERWRK.step0_12();
    title3 "data=INTERWRK.step0_12()";
    id sein year quarter seinunit;
    var wages_202 emp1_202 emp2_202 emp3_202 emp1_UI emp2_UI emp3_UI
wages_UI total_wages empl_month1 empl_month2 empl_month3
best_wages best_emp1 best_emp2 best_emp3
sein year quarter seinunit;
run;

/*
* proc contents data=INTERWRK.ecf_&state._leg_structure;
* run;
*/

```

QA INSTRUCTIONS: Special_handle can be positive when structure_fix is zero. this case represents candidate records that could not be fixed .

```

proc freq data=INTERWRK.&state._employer_char_unit;
  tables yr_qtr / missing;
  tables structure_fix;
  tables best_flag*structure_fix /v5fmt missing;
  tables special_handle*structure_fix /v5fmt missing;
run;

proc univariate data=INTERWRK.&state._employer_char_unit;
  var best_wages best_emp1 best_emp2 best_emp3;
run;

proc print data=INTERWRK.&state._employer_char_unit(obs=1000);
run;

proc print data=INTERWRK.&state._employer_char_unit(where=(structure_fix=1));
title3 "Structure Fix Listing";
run;

proc freq data=INTERWRK.&state._employer_char_unit;
  tables sic_invalid naics_1997_invalid naics_2002_invalid
naics_aux_1997_invalid naics_aux_2002_invalid;
  tables sic naics1997 naics2002 naics_aux1997 naics_aux2002 num_estabs county;
  tables owner_code best_flag ein_bad ein_defect valid_ein;
  tables ever_202 ever_UI ever_emp1 ever_emp2 ever_emp3 ever_wages ever_multi;
  tables in_202 in_UI multi_first_quarter multi_first_year multi_unit;
  tables seinunit_bad seinunit_type source special_handle structure_fix;
run;

proc means data=INTERWRK.&state._employer_char_unit n sum mean min max;
  class year quarter;
  title3 "Means of SEIN YEAR QUARTER SEINUNIT file";
  var best_emp1 best_emp2 best_emp3 best_wages;
  var empl_month1 empl_month2 empl_month3 total_wages;
run;

proc means data=INTERWRK.&state._employer_char_unit n sum mean min max;
  var emp1_UI emp2_UI emp3_UI wages_UI;
  var best_emp1 best_emp2 best_emp3 best_wages;
  var empl_month1 empl_month2 empl_month3 total_wages;
  var sein_emp1 sein_emp2 sein_emp3 sein_wages;
  var payroll seinsize_m seinsize_b seinsize_e;
  var sein_best_emp1 sein_best_emp2 sein_best_emp3 sein_best_wages;
run;

%end;
%mend;

```

1.5.3.20 library/macros/ecfdiag_13.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_13.sas >--+ */
/*----- LEGACY NAME: -----*/

```

This macro contains the diagnostic output formerly contained in . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_13(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_13 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 13";
/* see QA module 01 for diagnostics */

%end;
%mend;

```

1.5.3.21 library/macros/ecfdiag_14.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/ecfdiag_14.sas >--- */
/*----- LEGACY NAME: -----*/

```

This macro contains the diagnostic output formerly contained in . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_14(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_14 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 14";
/* see QA module 02 for diagnostics */

%end;
%mend;

```

1.5.3.22 library/macros/ecfdiag_15.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/ecfdiag_15.sas >--- */
/*----- LEGACY NAME: -----*/

```

This macro contains the diagnostic output formerly contained in . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_15(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_15 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 15";

/* see QA module 3 for the diagnostics here */

%end;
%mend;

```

1.5.3.23 library/macros/ecfdiag_16.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_16.sas >--+ */
/*----- LEGACY NAME: -----*/
```

This macro contains the diagnostic output formerly contained in . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```
%macro ecfdiag_16(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_16 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

Title2 "Module 16";

%end;
%mend;
```

1.5.3.24 library/macros/ecfdiag_17.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_17.sas >--+ */
/*----- LEGACY NAME: 17_naics_ldb_clean.sas -----*/
```

This macro contains the diagnostic output formerly contained in 17_naics_ldb_clean.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_17(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_17 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 17";

/** DIAGNOSTIC OUTPUT **/

proc contents data=INTERWRK.temp5_17;
run;

proc freq data=INTERWRK.temp5_17;
  tables merge candidate1997 candidate2002
  naics_ldb_1997_invalid*naics_ldb_1997_valid
  naics_ldb_2002_invalid*naics_ldb_2002_valid;
run;

proc freq data=INTERWRK.temp5_17(where=(naics_ldb_1997_invalid~="0"));
  tables naics_ldb1997_clean*naics_ldb_1997_invalid;
run;

proc freq data=INTERWRK.temp5_17(where=(naics_ldb_2002_invalid~="0"));
  tables naics_ldb2002_clean*naics_ldb_2002_invalid;
run;

proc print data=INTERWRK.temp5_17(where=(candidate1997=1));
  var sein seinunit naics_ldb_1997_invalid naics_ldb_1997_valid naics_ldb1997
  naics_ldb1997_clean naics_ldb1997_impute;
run;

proc print data=INTERWRK.temp5_17(where=(candidate2002=1));
  var sein seinunit naics_ldb_2002_invalid naics_ldb_2002_valid naics_ldb2002
  naics_ldb2002_clean naics_ldb2002_impute;
run;

/* AFTER: Reattach the cleaned NAICS codes back to the original */

proc contents data=INTERWRK.naics_ldb_clean_17;
run;

proc freq data=INTERWRK.naics_ldb_clean_17;
  tables merge;
run;

proc print data=INTERWRK.naics_ldb_clean_17(obs=1000);
run;

%end;
%mend;

```

1.5.3.25 library/macros/ecfdiag_18.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_18.sas >--+ */
/*----- LEGACY NAME: 18_seinunit_wide.sas -----*/

```

This macro contains the diagnostic output formerly contained in 18_seinunit_wide.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_18(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_18 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 18";

proc contents data=INTERWRK.seinunit_wide_18;
run;

proc print data=INTERWRK.seinunit_wide_18(obs=1000);
run;

%end;
%mend;

```

1.5.3.26 library/macros/ecfdiag_19.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_19.sas >--+ */
/*----- LEGACY NAME: 19_seinunit_fill.sas -----*/

```

This macro contains the diagnostic output formerly contained in 19_seinunit_fill.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_19(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_19 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 19";

```

Diagnostics.

```

proc contents data=INTERWRK.seinunit_long_19;
run;

proc freq data=INTERWRK.seinunit_long_19;
  tables es_sic_miss es_sic_flag /v5fmt;
  tables es_naics1997_miss es_naics1997_flag /v5fmt;
  tables es_naics2002_miss es_naics2002_flag /v5fmt;
  tables es_naics_aux1997_miss es_naics_aux1997_flag /v5fmt;
  tables es_naics_aux2002_miss es_naics_aux2002_flag /v5fmt;
  tables es_naics_ldb1997_miss es_naics_ldb1997_flag /v5fmt;
  tables es_naics_ldb2002_miss es_naics_ldb2002_flag /v5fmt;
  tables es_county_miss es_county_flag /v5fmt;
  tables es_owner_code_miss es_owner_code_flag /v5fmt;
  tables es_ein_miss es_ein_flag /v5fmt;
  tables es_sic_miss*es_naics1997_miss /v5fmt;
  tables es_sic_miss*es_naics_aux1997_miss /v5fmt;
  tables es_sic_miss*es_naics_ldb1997_miss /v5fmt;
  tables es_sic_miss*es_naics2002_miss /v5fmt;
  tables es_sic_miss*es_naics_aux2002_miss /v5fmt;
  tables es_sic_miss*es_naics_ldb2002_miss /v5fmt;
run;

proc print data=INTERWRK.seinunit_long_19(obs=1000);
id sein seinunit year quarter;
run;

%end;
%mend;

```

1.5.3.27 library/macros/ecfdiag_20.sas

```

/* +---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* +---< Location: /programs/production/dev1/current/ecf >---+ */
/* +---< File: library/macros/ecfdiag_20.sas >---+ */
/*----- LEGACY NAME: 20_seinunit_sync.sas -----*/

```

This macro contains the diagnostic output formerly contained in 20_seinunit_sync.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_20(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_20 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 20";

```

Diagnostics.

```

proc contents data=INTERWRK.seinunit_long_20;
run;
proc print data=INTERWRK.seinunit_long_20(obs=100);
run;

proc freq data=INTERWRK.seinunit_long_20;
  tables merge;
  tables es_sic_miss*any_valid /v5fmt missing;
  tables es_sic_src*any_valid /v5fmt missing;

  tables es_naics1997_miss*any_valid /v5fmt missing;
  tables es_naics1997_src*any_valid /v5fmt missing;
  tables es_naics_aux1997_miss*any_valid /v5fmt missing;
  tables es_naics_aux1997_src*any_valid /v5fmt missing;
  tables es_naics_ldb1997_miss*any_valid /v5fmt missing;
  tables es_naics_ldb1997_src*any_valid /v5fmt missing;
  tables es_naics_fnl1997_miss*any_valid /v5fmt missing;
  tables es_naics_fnl1997_src*any_valid /v5fmt missing;
  tables es_naics_imp1997_miss*any_valid /v5fmt missing;
  tables es_naics_imp1997_src*any_valid /v5fmt missing;
  tables es_naics_eso1997_miss*any_valid /v5fmt missing;
  tables es_naics_eso1997_src*any_valid /v5fmt missing;

  tables es_naics2002_miss*any_valid /v5fmt missing;
  tables es_naics2002_src*any_valid /v5fmt missing;
  tables es_naics_aux2002_miss*any_valid /v5fmt missing;
  tables es_naics_aux2002_src*any_valid /v5fmt missing;
  tables es_naics_ldb2002_miss*any_valid /v5fmt missing;
  tables es_naics_ldb2002_src*any_valid /v5fmt missing;
  tables es_naics_fnl2002_miss*any_valid /v5fmt missing;
  tables es_naics_fnl2002_src*any_valid /v5fmt missing;
  tables es_naics_imp2002_miss*any_valid /v5fmt missing;
  tables es_naics_imp2002_src*any_valid /v5fmt missing;
  tables es_naics_eso2002_miss*any_valid /v5fmt missing;
  tables es_naics_eso2002_src*any_valid /v5fmt missing;

  tables es_naics1997_miss*es_naics1997_src /v5fmt missing;
  tables es_naics2002_miss*es_naics2002_src /v5fmt missing;
  tables es_naics_aux1997_miss*es_naics_aux1997_src /v5fmt missing;
  tables es_naics_aux2002_miss*es_naics_aux2002_src /v5fmt missing;
  tables es_naics_ldb1997_miss*es_naics_ldb1997_src /v5fmt missing;
  tables es_naics_ldb2002_miss*es_naics_ldb2002_src /v5fmt missing;
  tables es_naics_fnl1997_miss*es_naics_fnl1997_src /v5fmt missing;
  tables es_naics_fnl2002_miss*es_naics_fnl2002_src /v5fmt missing;
  tables es_naics_imp1997_miss*es_naics_imp1997_src /v5fmt missing;
  tables es_naics_imp2002_miss*es_naics_imp2002_src /v5fmt missing;
  tables es_naics_eso1997_miss*es_naics_eso1997_src /v5fmt missing;
  tables es_naics_eso2002_miss*es_naics_eso2002_src /v5fmt missing;
  tables es_sic_miss*es_sic_src /v5fmt missing;

```

```

tables year*es_naics1997_src /v5fmt missing;
tables year*es_naics2002_src /v5fmt missing;
tables year*es_naics_aux1997_src /v5fmt missing;
tables year*es_naics_aux2002_src /v5fmt missing;
tables year*es_naics_ldb1997_src /v5fmt missing;
tables year*es_naics_ldb2002_src /v5fmt missing;
tables year*es_naics_fnl1997_src /v5fmt missing;
tables year*es_naics_fnl2002_src /v5fmt missing;
tables year*es_naics_imp1997_src /v5fmt missing;
tables year*es_naics_imp2002_src /v5fmt missing;
tables year*es_naics_eso1997_src /v5fmt missing;
tables year*es_naics_eso2002_src /v5fmt missing;
tables year*es_sic_src /v5fmt missing;
tables year*es_sic_src /v5fmt missing;

tables year*es_naics1997_miss /v5fmt missing;
tables year*es_naics2002_miss /v5fmt missing;
tables year*es_naics_aux1997_miss /v5fmt missing;
tables year*es_naics_aux2002_miss /v5fmt missing;
tables year*es_naics_ldb1997_miss /v5fmt missing;
tables year*es_naics_ldb2002_miss /v5fmt missing;
tables year*es_naics_fnl1997_miss /v5fmt missing;
tables year*es_naics_fnl2002_miss /v5fmt missing;
tables year*es_naics_imp1997_miss /v5fmt missing;
tables year*es_naics_imp2002_miss /v5fmt missing;
tables year*es_naics_eso1997_miss /v5fmt missing;
tables year*es_naics_eso2002_miss /v5fmt missing;
tables year*es_sic_miss /v5fmt missing;
tables year*es_sic_miss /v5fmt missing;

tables es_naics1997*es_naics1997_miss /v5fmt missing;
tables es_naics_aux1997*es_naics_aux1997_miss /v5fmt missing;
tables es_naics_ldb1997*es_naics_ldb1997_miss /v5fmt missing;
tables es_naics_fnl1997*es_naics_fnl1997_miss /v5fmt missing;
tables es_naics_imp1997*es_naics_imp1997_miss /v5fmt missing;
tables es_naics_eso1997*es_naics_eso1997_miss /v5fmt missing;

tables es_naics1997*es_naics1997_miss /v5fmt missing;
tables es_naics_aux2002*es_naics_aux2002_miss /v5fmt missing;
tables es_naics_ldb2002*es_naics_ldb2002_miss /v5fmt missing;
tables es_naics_fnl2002*es_naics_fnl2002_miss /v5fmt missing;
tables es_naics_imp2002*es_naics_imp2002_miss /v5fmt missing;
tables es_naics_eso2002*es_naics_eso2002_miss /v5fmt missing;
run;

%end;
%mend;

```

1.5.3.28 library/macros/ecfdiag_21.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--> */
/* +--< Location: /programs/production/dev1/current/ecf >--> */
/* +--< File: library/macros/ecfdiag_21.sas >--> */
/*----- LEGACY NAME: 21_sein_mode_calc.sas -----*/

```

This macro contains the diagnostic output formerly contained in 21_sein_mode_calc.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_21(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_21 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 21";

```

Diagnostics.

```

proc print data=INTERWRK.sein_modes_21(obs=111);
run;

proc freq data=INTERWRK.sein_modes_21;
  tables mode_es_sic mode_es_county mode_es_owner_code /v5fmt;
  tables mode_es_naics_eso1997 mode_es_naics_eso2002 /v5fmt;
  tables mode_es_naics_fnl1997 mode_es_naics_fnl2002 /v5fmt;
  tables mode_leg_wib mode_leg_msapmsa mode_leg_state mode_leg_county
mode_leg_subctygeo /v5fmt;
  tables mode_es_sic_emp mode_es_county_emp mode_es_owner_code_emp /v5fmt;
  tables mode_es_naics_eso1997_emp mode_es_naics_eso2002_emp /v5fmt;
  tables mode_es_naics_fnl1997_emp mode_es_naics_fnl2002_emp /v5fmt;
  tables mode_leg_wib_emp mode_leg_msapmsa_emp mode_leg_state_emp
mode_leg_county_emp mode_leg_subctygeo_emp /v5fmt;
run;

%end;
%mend;

```

1.5.3.29 library/macros/ecfdiag_22.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/ecfdiag_22.sas >--- */
/*----- LEGACY NAME: 22_sein_wide.sas -----*/

```

This macro contains the diagnostic output formerly contained in 22_sein_wide.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_22(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_22 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

Title2 "Module 22";

```

Diagnostics

```

proc contents data=INTERWRK.sein_wide_22;
run;

proc print data=INTERWRK.sein_wide_22(obs=1000);
run;

proc contents data=INTERWRK.sein_long_22;
run;

proc print data=INTERWRK.sein_long_22(obs=1000);
run;

%end;
%mend;

```

1.5.3.30 library/macros/ecfdiag_23.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_23.sas >--+ */
/*----- LEGACY NAME: 23_sein_fill.sas -----*/

```

This macro contains the diagnostic output formerly contained in 23_sein_fill.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_23(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_23 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

Title2 "Module 23";

```

Diagnostics.

```

proc contents data=INTERWRK.sein_mode_fill_23;
run;

proc freq data=INTERWRK.sein_mode_fill_23;
  tables mode_es_sic_emp_miss mode_es_sic_emp_flag /v5fmt;
  tables mode_es_naics_eso1997_emp_miss mode_es_naics_eso1997_emp_flag /v5fmt;
  tables mode_es_naics_eso2002_emp_miss mode_es_naics_eso2002_emp_flag /v5fmt;
  tables mode_es_naics_fnl1997_emp_miss mode_es_naics_fnl1997_emp_flag /v5fmt;
  tables mode_es_naics_fnl2002_emp_miss mode_es_naics_fnl2002_emp_flag /v5fmt;
  tables mode_es_county_emp_miss mode_es_county_emp_flag /v5fmt;
  tables mode_es_owner_code_emp_miss mode_es_owner_code_emp_flag /v5fmt;
  tables mode_es_ein_emp_miss mode_es_ein_emp_flag /v5fmt;
  tables mode_leg_wib_emp_miss mode_leg_wib_emp_flag /v5fmt;
  tables mode_leg_msapmsa_emp_miss mode_leg_msapmsa_emp_flag /v5fmt;
  tables mode_leg_state_emp_miss mode_leg_state_emp_flag /v5fmt;
  tables mode_leg_county_emp_miss mode_leg_county_emp_flag /v5fmt;
  tables mode_leg_subctygeo_emp_miss mode_leg_subctygeo_emp_flag /v5fmt;
run;

proc freq data=INTERWRK.sein_mode_fill_23;
  tables mode_es_sic_miss mode_es_sic_flag /v5fmt;
  tables mode_es_naics_eso1997_miss mode_es_naics_eso1997_flag /v5fmt;
  tables mode_es_naics_eso2002_miss mode_es_naics_eso2002_flag /v5fmt;
  tables mode_es_naics_fnl1997_miss mode_es_naics_fnl1997_flag /v5fmt;
  tables mode_es_naics_fnl2002_miss mode_es_naics_fnl2002_flag /v5fmt;
  tables mode_es_county_miss mode_es_county_flag /v5fmt;
  tables mode_es_owner_code_miss mode_es_owner_code_flag /v5fmt;
  tables mode_es_ein_miss mode_es_ein_flag /v5fmt;
  tables mode_leg_wib_miss mode_leg_wib_flag /v5fmt;
  tables mode_leg_msapmsa_miss mode_leg_msapmsa_flag /v5fmt;
  tables mode_leg_state_miss mode_leg_state_flag /v5fmt;
  tables mode_leg_county_miss mode_leg_county_flag /v5fmt;
  tables mode_leg_subctygeo_miss mode_leg_subctygeo_flag /v5fmt;
run;

proc print data=INTERWRK.sein_mode_fill_23(obs=1000);
id sein year quarter;
run;

proc contents data=INTERWRK.sein_missing_23;
run;

proc print data=INTERWRK.sein_missing_23(obs=1000);
run;

%end;
%mend;

```

1.5.3.31 library/macros/ecfdiag_24.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_24.sas >--+ */
/*----- LEGACY NAME: 24_impute_sein_industry.sas -----*/

```

This macro contains the diagnostic output formerly contained in 24_impute_sein_industry.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_24(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_24 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

Title2 "Module 24";

proc contents data=INTERWRK.sicfreq_24;
run;

proc print data=INTERWRK.sicfreq_24;
run;

proc print data=INTERWRK.sein_missing_24(obs=1000);
  id sein;
  var uniform0 sic_impute;
run;

proc freq data=INTERWRK.sein_missing_24;
  tables sic_impute;
run;

proc contents data=INTERWRK.sein_mode_fill_23;
run;

proc print data=INTERWRK.sein_mode_fill_23(obs=1000);
  title3 "INPUT Data";
run;

```

```

proc contents data=INTERWRK.sein_long_24;
run;

proc freq data=INTERWRK.sein_long_24;
  title3 "Output Data";
  tables _merge mode_es_sic*mode_es_sic_miss
         mode_es_sic_emp*mode_es_sic_emp_miss
         /nocol norow nopercent;
  tables mode_es_naics_fnl1997*mode_es_naics_fnl1997_miss
         mode_es_naics_fnl1997_emp*mode_es_naics_fnl1997_emp_miss
         /nocol norow nopercent;
  tables mode_es_naics_fnl2002*mode_es_naics_fnl2002_miss
         mode_es_naics_fnl2002_emp*mode_es_naics_fnl2002_emp_miss
         /nocol norow nopercent;
  tables mode_es_naics_eso1997*mode_es_naics_eso1997_miss
         mode_es_naics_eso1997_emp*mode_es_naics_eso1997_emp_miss
         /nocol norow nopercent;
  tables mode_es_naics_eso2002*mode_es_naics_eso2002_miss
         mode_es_naics_eso2002_emp*mode_es_naics_eso2002_emp_miss
         /nocol norow nopercent;
  tables mode_es_naics_fnl1997_emp_miss mode_es_naics_fnl2002_emp_miss
         mode_es_naics_eso1997_emp_miss mode_es_naics_eso2002_emp_miss;
  tables mode_es_county_emp_miss mode_es_owner_code_emp_miss
         mode_es_ein_emp_miss;
  tables mode_es_county_miss mode_es_owner_code_miss mode_es_ein_miss;
run;

proc print data=INTERWRK.sein_long_24(obs=1000);
  title3 "Output Data";
run;

%end;
%mend;

```

1.5.3.32 library/macros/ecfdiag_25.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_25.sas >--+ */
/*----- LEGACY NAME: 25_impute_seinunit_industry.sas -----*/

```

This macro contains the diagnostic output formerly contained in 25_impute_seinunit_industry.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_25(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_25 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

Title2 "Module 25";

proc contents data=INTERWRK.seinunit_long_25;
run;

proc freq data=INTERWRK.seinunit_long_25;
  tables merge es_sic_miss es_sic;
  tables es_naics_fnl1997_miss es_naics_fnl1997 es_naics_fnl2002_miss
es_naics_fnl2002;
  tables es_naics_eso1997_miss es_naics_eso1997 es_naics_eso2002_miss
es_naics_eso2002;
  tables es_county_miss es_county es_owner_code_miss es_owner_code es_ein_miss;
run;

%end;
%mend;

```

1.5.3.33 library/macros/ecfdiag_26.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/ecfdiag_26.sas >--- */
/*----- LEGACY NAME: 26_seinunit_yq_chars.sas -----*/

```

This macro contains the diagnostic output formerly contained in 26_seinunit_yq_chars.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_26(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_26 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Module 26";

```

Diagnostics.

```

proc contents data=INTERWRK.&state._employer_char_unit3();
run;

proc freq data=INTERWRK.&state._employer_char_unit3();
  tables _merge;
  tables yr_qtr /v5fmt;
  tables yr_qtr*source /v5fmt;
run;

proc freq data=INTERWRK.&state._employer_char_unit3();
  tables num_estabs;
  tables source
    in_202*source
    in_UI*source
    structure_fix
    special_handle;
  tables multi_first_quarter
    multi_first_year
    multi_unit_code*multi_unit;
  tables ever_202
    ever_UI
    ever_emp1
    ever_emp2
    ever_emp3
    ever_wages
    ever_multi;
  tables seinunit_bad
    seinunit_type
    es_owner_code
    auxiliary_code;
  tables es_county
    leg_county
    /v5fmt missing;
  tables leg_county
    leg_flag_geo
    leg_geo_qual
    leg_msapmsa
    leg_state
    leg_wib;
  tables es_county_flag
    es_county_miss
    es_ein_flag
    es_ein_miss
    es_owner_code_flag
    es_owner_code_miss
    es_sic_flag
    es_sic_miss;

```

```

tables es_sic es_sic_2
      es_sic_3
      es_sic_div
      /v5fmt missing;
tables es_naics_eso1997
      es_naics_eso2002
      /v5fmt missing;
tables es_naics_fnl1997
      es_naics_fnl1997_2
      es_naics_fnl1997_3
      es_naics_fnl1997_4
      es_naics_fnl1997_5
      /v5fmt missing;
tables es_naics_fnl2002
      es_naics_fnl2002_2
      es_naics_fnl2002_3
      es_naics_fnl2002_4
      es_naics_fnl2002_5
      /v5fmt missing;
tables es_naics_eso1997_miss
      es_naics_eso2002_miss
      es_naics_eso1997_src
      es_naics_eso2002_src;
tables es_naics_fnl1997_miss
      es_naics_fnl2002_miss
      es_naics_fnl1997_src
      es_naics_fnl2002_src;
tables mode_es_sic
      mode_es_naics_fnl1997
      mode_es_naics_fnl2002
      mode_es_county
      mode_es_owner_code
      /v5fmt;
tables mode_leg_wib
      mode_leg_msapmsa
      mode_leg_state
      mode_leg_county
      mode_leg_subctygeo
      /v5fmt;
tables mode_es_sic_emp
      mode_es_naics_fnl1997_emp
      mode_es_naics_fnl2002_emp
      mode_es_county_emp
      mode_es_owner_code_emp
      /v5fmt;
tables mode_leg_wib_emp
      mode_leg_msapmsa_emp
      mode_leg_state_emp
      mode_leg_county_emp
      mode_leg_subctygeo_emp
      /v5fmt;

```

```

tables mode_es_county_emp_flag
      mode_es_county_emp_miss
      mode_es_county_flag
      mode_es_county_miss ;
tables mode_es_ein_emp_flag
      mode_es_ein_emp_miss
      mode_es_ein_flag
      mode_es_ein_miss ;
tables mode_es_naics_eso1997_emp_flag
      mode_es_naics_eso1997_emp_miss
      mode_es_naics_eso1997_flag
      mode_es_naics_eso1997_miss;
tables mode_es_naics_eso2002_emp_flag
      mode_es_naics_eso2002_emp_miss
      mode_es_naics_eso2002_flag
      mode_es_naics_eso2002_miss ;
tables mode_es_naics_fnl1997_emp_flag
      mode_es_naics_fnl1997_emp_miss
      mode_es_naics_fnl1997_flag
      mode_es_naics_fnl1997_miss ;
tables mode_es_naics_fnl2002_emp_flag
      mode_es_naics_fnl2002_emp_miss
      mode_es_naics_fnl2002_flag
      mode_es_naics_fnl2002_miss ;
tables mode_es_owner_code_emp_flag
      mode_es_owner_code_emp_miss
      mode_es_owner_code_flag
      mode_es_owner_code_miss ;
tables mode_es_sic_emp_flag
      mode_es_sic_emp_miss
      mode_es_sic_flag
      mode_es_sic_miss ;
tables mode_leg_state_emp_flag
      mode_leg_state_emp_miss
      mode_leg_state_flag
      mode_leg_state_miss ;
tables mode_leg_county_emp_flag
      mode_leg_county_emp_miss
      mode_leg_county_flag
      mode_leg_county_miss ;
tables mode_leg_msapmsa_emp_flag
      mode_leg_msapmsa_emp_miss
      mode_leg_msapmsa_flag
      mode_leg_msapmsa_miss ;
tables mode_leg_subctygeo_emp_flag
      mode_leg_subctygeo_emp_miss
      mode_leg_subctygeo_flag
      mode_leg_subctygeo_miss ;

tables mode_leg_wib_emp_flag
      mode_leg_wib_emp_miss
      mode_leg_wib_flag
      mode_leg_wib_miss ;
tables master_empl_month1_flg
      master_empl_month2_flg
      master_empl_month3_flg;
tables master_total_wages_flg
      master_multi_unit_code*multi_unit_code;
run;

```

```

proc means data=INTERWRK.&state._employer_char_unit3();
  var emp1_UI emp2_UI emp3_UI wages_UI;
  var best_emp1 best_emp2 best_emp3 best_wages;
  var empl_month1 empl_month2 empl_month3 total_wages;
  var sein_emp1 sein_emp2 sein_emp3 sein_wages;
  var payroll seinsize_m seinsize_b seinsize_e;
  var sein_best_emp1 sein_best_emp2 sein_best_emp3 sein_best_wages;
run;

proc print data=INTERWRK.&state._employer_char_unit3(obs=1000);
id sein year quarter seinunit;
run;

%end;
%mend;

```

1.5.3.34 library/macros/ecfdiag_27.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_27.sas >--+ */
/*----- LEGACY NAME: 27_weight_calc_01.sas -----*/

```

This macro contains the diagnostic output formerly contained in 27_weight_calc.01.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_27(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_27 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

Title2 "Module 27: Creation of the weights, step 1."
proc contents data=INTERWRK.weights_sein_27;
run;

proc means data=INTERWRK.weights_sein_27;
run;

proc print data=INTERWRK.weights_sein_27(obs=1000);
run;

%end;
%mend;

```

1.5.3.35 library/macros/ecfdiag_28.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_28.sas >--+ */
/*----- LEGACY NAME: 28_weight_calc_02.sas -----*/

```

This macro contains the diagnostic output formerly contained in 28_weight_calc.02.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_28(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_28 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

```

Diagnostics.

```

title2 "Diagnostics: Module 28";

proc means data=INTERWRK.weights_sein_27(where=(get_weight=1 and seinsize_b>0))
n sum ;
  title3 "sums for get_weight=1 and seinsize_b>0";
  class year quarter;
  var sein_best_emp1 sein_best_emp1_adjust sein_best_emp1_weight emp1_w2
emp1_w2_trim emp1_UI seinsize_b;
run;

```

Diagnostics

```

proc print data=INTERWRK.weight_totals_&state._28;
  title3 "Final totals dataset with BLS and ECF";
run;

```

Diagnostics

```

proc contents data=INTERWRK.weights_sein_28;
run;

proc freq INTERWRK.weights_sein_28;
  title3 "After merge with totals data";
  tables merge;
  tables yr_qtr*get_weight;
run;

proc means data=INTERWRK.weights_sein_28 n sum;
  title1 "All Records";
  class year quarter;
  var emp1_UI_weight emp1_UI_notrim emp1_UI_adjust b_adjust qwi_unit_weight;
run;

proc means data=INTERWRK.weights_sein_28(where=(count_inscope>0)) n sum;
  title1 "Records where b>0";
  class year quarter;
  var emp1_UI_weight emp1_UI_notrim emp1_UI_adjust b_adjust qwi_unit_weight;
run;

proc univariate data=INTERWRK.weights_sein_28(where=(get_weight=1));
  class yr_qtr rectype;
  var w2 qwi_unit_weight;
run;

proc print data=INTERWRK.weights_sein_28(obs=1000);
  id sein year quarter;
run;

```

Diagnostics

```

proc contents data=INTERWRK.&state._employer_char_unit4; ;
run;

proc freq data=INTERWRK.&state._employer_char_unit4;
  title3 " Merge the weights on before adding the fuzz.";
  tables merge;
run;

proc print data=INTERWRK.&state._employer_char_unit4(obs=1000);
  id sein year quarter;
  var in_UI in_202 ever_202 qwi_unit_weight;
run;

%end;
%mend;

```

1.5.3.36 library/macros/ecfdiag_29.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_29.sas >--+ */
/*----- LEGACY NAME: 29_seinunit_fuzz.sas -----*/

```

This macro contains the diagnostic output formerly contained in 29_seinunit_fuzz.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_29(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_29 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Diagnostics: Module 29";

```

Diagnostic output.

```

proc contents data=OUTPUTS.ecf_&state._seinunit;
run;

proc contents data=INTERWRK.&state._master_sein_fuzz;
proc contents data=INTERWRK.new_sein_fuzz_29;
run;

proc contents data=INTERWRK.&state._master_seinunit_fuzz;
proc contents data=INTERWRK.new_seinunit_fuzz_29;
run;

```

Print univariate of actual distribution and example values of fuzz factors.

```

/*BEGINCONFIDENTIAL=====*/
proc univariate data=OUTPUTS.ecf_&state._seinunit;
  var ramp_sein ramp_seinunit beta_sein beta_seinunit;
run;

proc print data=OUTPUTS.ecf_&state._seinunit(obs=200);
title3 "Fuzz Factors: SEIN YEAR QUARTER SEINUNIT";
var sein year quarter seinunit RAMP_SEIN RAMP_SEINUNIT BETA_SEIN BETA_SEINUNIT;
run;

proc print data=INTERWRK.&state._master_sein_fuzz(obs=200);
title3 "Fuzz Factors: SEIN";
var sein RAMP_SEIN BETA_SEIN;
run;

proc print data=INTERWRK.&state._master_seinunit_fuzz(obs=300);
title3 "Fuzz Factors: SEIN SEINUNIT";
var sein seinunit RAMP_SEINUNIT BETA_SEINUNIT;
run;
/*=====CONFIDENTIALEND*/

```

Graph the Fuzz Factors. This should be part of the QA.

```

filename graph1 "./29_seinunit_fuzz.cgm";

goptions device=cgmof971 gsfname=graph1 gsfmode=replace;

/*BEGINCONFIDENTIAL=====*/
proc univariate data=OUTPUTS.ecf_&state._seinunit;
  var ramp_sein ramp_seinunit beta_sein beta_seinunit;
  histogram ramp_sein ramp_seinunit beta_sein beta_seinunit / midpoints=.8 to
1.2 by .0005;
run;
/*=====CONFIDENTIALEND*/

%end;
%mend;

```

1.5.3.37 library/macros/ecfdiag_30.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/ecfdiag_30.sas >--- */
/*----- LEGACY NAME: 30_sein_file.sas -----*/

```

This macro contains the diagnostic output formerly contained in 30_sein_file.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

```

%macro ecfdiag_30(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_30 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Diagnostics: Module 30";

```

Diagnostics.

```
proc contents data=OUTPUTS.ecf_&state._sein;
run;

proc means data=OUTPUTS.ecf_&state._sein;;
run;

proc print data=OUTPUTS.ecf_&state._sein(obs=1000);
run;
```

A QA here could entail comparing number of unique SEINs on both the SEIN and the SEINUNIT file. Should be trivially satisfied.

```
%end;
%mend;
```

1.5.3.38 library/macros/ecfdiag_31.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/ecfdiag_31.sas >--+ */
/*----- LEGACY NAME: 31_fuzz_update.sas -----*/
```

This macro contains the diagnostic output formerly contained in 31_fuzz_update.sas . It is processed conditional on the parameter &diagnostics , which is set in the parameters file.

Diagnostics and QA on merge. Instructions: check the merge_check file. If no 3s, then OK, otherwise critical error. This is already checked in QA module 05?

```

%macro ecfdiag_31(doit);
%if ( "&doit" = "" ) %then %let doit=no;
%put %upcase(note): Processing ecfdiag_31 : &doit.;
%if ( "&doit" = "YES" or "&doit" = "yes" ) %then %do;

title2 "Diagnostics: Module 31";

proc contents data=OUTPUTS.ecf_&state._seinunit_fuzz;
run;

proc freq data=INTERWRK.fuzz_merge_check2;
tables merge;
run;

proc means data=OUTPUTS.ecf_&state._seinunit_fuzz;
run;

proc freq data=OUTPUTS.ecf_&state._seinunit_fuzz;
format date_seinunit_fuzz MMDDYYS10.;
tables UPDATE_NUMBER_SEINUNIT date_seinunit_fuzz;
run;

proc print data=OUTPUTS.ecf_&state._seinunit_fuzz(obs=100);
format date_seinunit_fuzz MMDDYYS10.;
id sein seinunit;
run;

%end;
%mend;

```

1.5.3.39 library/macros/ecfqa_checkfuzz.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/ecfqa_checkfuzz.sas >---+ */
/* Time-stamp: <04/09/02 23:07:40 vilhuber> */

```

This defines a macro to analyze the Fuzz factor merges. Input and output dataset will have the same names, but input is expected in WORK, output will be written to INTERWRK.

```

%macro ecfqa_checkfuzz(inoutdata=);

data INTERWRK.&inoutdata.;
merge
  &inoutdata. (in=a)
  merge_codes(in=b);
by merge;
length result $ 5 note $ 80;
note="";
result="OK";
if count = . then count = 0;

```

If codes are both on the new and the old file after the merge (flagged by a value of 3), then some SEIN or SEINUNIT was re-assigned a new code. This is a FATAL error, and should induce a full stop to processing. It should never occur.

```

if merge = 3 and count > 0 then do;
note="Fatal error in processing: error in Fuzz merge";
result="%upcase(error)";
call symput('result',trim(left(result)));
call symput('note',trim(left(note)));
end;

```

If 100% of fuzz factors are new, it is either a new state (benign, and can pass), or a serious processing error occurred (critical failure).

```
    else if merge = 2 and percent = 100 then do;
note="100% new fuzz factors: critical error in Fuzz merge or new state";
result="%upcase(error)";
call symput('result',trim(left(result)));
call symput('note',trim(left(note)));
end;
```

If over 90 percent of fuzz factors are new (merge code 2) then some error occurred that made the previous fuzz factor file unavailable or corrupted. This is a critical error.

```
    else if merge = 2 and percent > 90 then do;
note="> 90% new fuzz factors: error in Fuzz merge";
result="%upcase(error)";
call symput('result',trim(left(result)));
call symput('note',trim(left(note)));
end;
```

If between 90 and 50 percent of fuzz factors are new (merge code 2) then some strange scenario occurred. This is less than on a new file, but more than typically could be expected.

```
    else if merge = 2 and percent > 50 then do;
note="Over 50% new fuzz factors: possible error in Fuzz merge";
result="%upcase(warning)";
call symput('result',trim(left(result)));
call symput('note',trim(left(note)));
end;
```

If NO new fuzz factors were assigned, then this data was processed through the ECF before, or some strange data configuration. The latter should obviously be investigated. The former scenario indicates incorrect processing, but is NOT a fatal error.

```
    else if merge = 2 and percent = 0 then do;
note="No new fuzz factors: Likely repeat processing of ECF? ";
result="%upcase(warning)";
call symput('result',trim(left(result)));
call symput('note',trim(left(note)));
end;
```

The last situation, when a very small positive number of fuzz factors has been assigned, is also indicative of a strange data configuration, but could occur for legitimate reasons.

```
    else if merge = 2 and percent < 1 then do;
note="Less than 1% new fuzz factors: possible error in Fuzz merge ";
result="%upcase(warning)";
call symput('result',trim(left(result)));
call symput('note',trim(left(note)));
end;
    else if merge=2 then call symput('note',trim(left(count))||" new fuzz
factors");
run;
%mend;
```

1.5.3.40 library/macros/fixfips.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/fixfips.sas >---+ */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/*-----*/
/**** MACRO to fix the 2-digit state problem ****/
/*-----*/

%macro fixfips;
%if %eval(&stfips<10) %then %do;
    %let stfips=0&stfips;
%end;
%mend;
```

1.5.3.41 library/macros/fzexist.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/fzexist.sas >---+ */
/* Time-stamp: <05/01/23 22:09:58 vilhuber> */
/*****
```

MACRO to test for the existence of the fuzz files. If the files do not exist then create empty ones in INTERWRK with the correct structure. If they do exist, copy the existing ones from INPUTS to INTERWRK, and change the label.

```
*****/
%macro fzexist(dset1,dset2);
    %if not(%sysfunc(exist(&dset1)) or %sysfunc(exist(&dset2))) %then %do;
```

Create a dummy SEIN fuzz dataset

```
data INTERWRK.&state._master_sein_fuzz
(label="Created on %sysfunc(date()),worddate.")
;

/*BEGINCONFIDENTIAL=====*/
length sein $12 BETA_SEIN RAMP_SEIN UNIFORM_SEIN 8;

label BETA_SEIN="Beta Fuzz Factor";
label RAMP_SEIN="Ramp Fuzz Factor";
label UNIFORM_SEIN="Draw from Uni Dist";
label DATE_SEIN_FUZZ="Creation Date of Fuzz Record";
label UPDATE_NUMBER_SEIN="Update Processing Counter";

sein=" ";
beta_sein=.;
ramp_sein=.;
uniform_sein=.;
/*=====CONFIDENTIALEND*/

date_sein_fuzz=.;
update_number_sein=.;
if sein~" " then output;
run;
```

Create a dummy SEINUNIT fuzz dataset

```

data INTERWRK.&state._master_seinunit_fuzz
  (label="Created on %sysfunc(date(),worddate.)")
;
/*BEGINCONFIDENTIAL====*/
  length sein $12 seinunit $5 BETA_SEINUNIT RAMP_SEINUNIT
UNIFORM_SEINUNIT 8;

  label BETA_SEINUNIT="Beta Fuzz Factor";
  label RAMP_SEINUNIT="Ramp Fuzz Factor";
  label UNIFORM_SEINUNIT="Draw from Uni Dist";
  label DATE_SEINUNIT_FUZZ="Creation Date of Fuzz Record";
  label UPDATE_NUMBER_SEINUNIT="Update Processing Counter";

  sein=" ";
  seinunit=" ";
  beta_seinunit=.;
  ramp_seinunit=.;
  uniform_seinunit=.;
/*====CONFIDENTIALEND*/

  date_seinunit_fuzz=.;
  update_number_seinunit=.;
  if sein~=" " then output;
run;
%end;
%else %do;

  data INTERWRK.&state._master_sein_fuzz
  (label="Last modified on %sysfunc(date(),worddate.)")
;
set &dset1.;
run;

  data INTERWRK.&state._master_seinunit_fuzz
  (label="Last modified on %sysfunc(date(),worddate.)")
;
set &dset2.;
run;

%end;
%mend;

```

1.5.3.42 library/macros/indlkup2.sas

```

/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/indlkup2.sas >--+ */
/* Time-stamp: <04/07/13 09:07:28 vilhuber> */
/*****

```

MACRO to look up NAICS/SIC for missing SIC/NAICS

```

/*****
/* mvar = missing variable or variable to impute */
/* lvar = variable used to do the impute or lookup */
/* mlen = missing variable length */
/* llen = lookup variable length */
/* kname= index name for lookup variable or input to impute */
/* mname= imputed value from lookup */
/* rvar = random variable name */
/* dset = dataset used for impute */

%macro indlkup2(mvar,lvar,mlen,llen,kname,mname,rvar,dset);
    length &mvar._clean &mname $&mlen &lvar._temp $&llen;
    lookup_temp=0;
    if &mvar._miss=1 and
(&lvar._miss=0 or &lvar._miss=1.5
    or &lvar._miss=2 or &lvar._miss=3 or &lvar._miss=4
    or &lvar._miss=5 or &lvar._miss=6)
then lookup_temp=1;
    %if &&lvar=es_sic or &&lvar=es_naics1997
or &&lvar=es_naics_aux1997
or &&lvar=es_naics_ldb1997 %then %do;

        if &mvar._miss=2 and year<2000 and sic_change>0
        then lookup_temp=2;
    %end;

```

```

if lookup_temp>0 then do;
  &lvar._temp=&lvar.;
  &kname=&lvar.;
  &mvar._clean=repeat("9",%eval(&mflen-1));

  found_ind_temp=0;
  continue=0;
  continue1=0;
  &mname=repeat("9",%eval(&mflen-1));
  do while ( continue=0 );
    set INPUTS.&dset key=&kname;
    if continue1=1 then continue=1;
    if _iorc_ = 0 then do;
      if &rvar>=low_limit and &rvar<=up_limit
then &mvar._clean=&mname.;
      %if &&lvar=es_sic
        or &&lvar=es_naics1997
      or &&lvar=es_naics_aux1997
        or &&lvar=es_naics_ldb1997
%then %do;
        if &mvar=&mname then found_ind_temp=1;
        %end;
      end;
    else do;
      _error_=0;
      &kname.=repeat("X",%eval(&lflen-1));
      continue1=1;
    end;
  end;

  *put sein year quarter seinunit &mvar= &lvar._temp=
  &mvar._miss= &lvar._miss= &mvar._clean= " &mvar &lvar";
  if lookup_temp=1 and &mvar._clean~=repeat("9",%eval(&mflen-1))
  then do;
    &mvar.=&mvar._clean;
    &mvar._miss=3;
  end;
  &lvar=&lvar._temp;
end;
%mend;

```

1.5.3.43 library/macros/indlkup.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/indlkup.sas >---+ */
/* Time-stamp: <01/15/04 10:26:50 mckin013> */
/*****
/* MACRO to look up SIC and NAICS codes by first X digits */
/*****
/* icode=index prefix */
/* var= variable prefix */
/* length= substr length for ind impute */
/* dset=impute dataset name */
/* rvar=random variable */

%macro indlkup(icode,var,length,dset,rvar);
  length &var.&length $&length;
  &var.&length=substr(&var,1,&length);
  continue=0;
  continue1=0;
  &var._impute=" ";
  do while ( continue=0 );
    set INPUTS.&dset key=&icode.&length;
    if continue1=1 then continue=1;
    if _iorc_ = 0 then do;
      if &rvar>=begin_value and &rvar<=end_value then
&var._clean=&var._impute;
      end;
    else do;
      _error_=0;
      &var.&length=repeat("X",%eval(&length-1));
      continue1=1;
    end;
  end;
end;
%mend;
```

1.5.3.44 library/macros/modevar.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/modevar.sas >--+ */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */

/*****
/** MACRO to calculate the Mode ***/
*****/

%macro modevar(datain,dataout,var,length,missval);
  data &dataout (keep=sein year quarter mode_&var mode_&var._emp);
  set &datain;
  by sein year quarter &var ;
  length mode_&var mode_&var._emp $&length;
  retain biggest biggest_emp &var._num &var._num_emp mode_&var
mode_&var._emp;
  if first.quarter then do;
    biggest=0;
    biggest_emp=0;
    mode_&var=&missval;
    mode_&var._emp=&missval;
  end;
  if first.&var then &var._num=0;
  if first.&var then &var._num_emp=0;
  &var._num+1;
  if best_emp1>0 or best_emp2>0 or best_emp3>0 then do;
    &var._num_emp+max(of best_emp1 best_emp2 best_emp3);
  end;
  if &var._num>biggest and &var ne &missval then do;
    biggest=&var._num;
    mode_&var=&var;
  end;
  if &var._num_emp>biggest_emp and &var ne &missval then do;
    biggest_emp=&var._num_emp;
    mode_&var._emp=&var;
  end;
  if last.quarter;
run;
title1 "Print of Mode &var";
proc print data=&dataout (obs=50);
run;
%mend;
```

1.5.3.45 library/macros/qloop.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/qloop.sas >---+ */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/*****
/* MACRO to loop over SEIN YEAR QUARTER in quarter time *****/
/*****

%macro qloop(varname,aryname,missval);
  %qtime(year,quarter) /* set qtime for current year and quarter*/;
  do i=1 to &qmax while(&varname=&missval);
    if &varname=&missval then do;
      if (qtime+i)<(&qend+1) then do;
        &varname._flag=i;
        qtloop=qtime+i;
        %yrqtr(qtloop);
        &varname=&aryname{yrtmp,qrtmp};
        *put sein= year= quarter= qtime= yrtmp= qrtmp=
&aryname{yrtmp,qrtmp}=;
        end;
      end;
      if &varname=&missval then do;
        if (qtime-i)>(&qbeg-1) then do;
          &varname._flag=-i;
          qtloop=qtime-i;
          %yrqtr(qtloop);
          &varname=&aryname{yrtmp,qrtmp};
          *put sein= year= quarter= qtime= yrtmp= qrtmp=
&aryname{yrtmp,qrtmp}=;
          end;
        end;
      end;
      if &varname=&missval then &varname._flag=.;
    %mend;

```

1.5.3.46 library/macros/qtime.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/qtime.sas >---+ */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/*****
*** MACROS to convert qtime to year,quarter and back *****/
/*****

/* FUNCTION TO CALCULATE QTIME for a given year quarter ***/
%macro qtime(year,quarter);
  qtime=(&year-&ymin)*4 + &quarter;
%mend;

```

1.5.3.47 library/macros/set_es.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/set_es.sas >---+ */
/* Time-stamp: <04/07/13 09:58:37 vilhuber> */
/*****
/* Create the set statement for the read in */
*****/
%macro set_es;
  set
  %do year=&start_year %to &end_year;
  %do q=1 %to 4;
  %if &year=&start_year and &q < &start_quarter %then %let q=&start_quarter;
  /* Create the set statement */
  INPUTS.es202_&state._&year.q&q.
  (* in=&year. */
  keep=sein year quarter seinunit owner_code
  ein county sic naics naics_aux multi_unit_code auxiliary_code

  empl_month1_flg empl_month2_flg empl_month3_flg total_wages_flg

  empl_month1 empl_month2 empl_month3 total_wages
  rename=(county=county_temp
  )
  %if &year=&end_year and &q = &end_quarter %then %let q=4;
  %end; /* end q loop */
  %end; /* end year loop */
%mend;
```

1.5.3.48 library/macros/set_ldb_qa.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/set_ldb_qa.sas >---+ */
/* Time-stamp: <04/09/17 19:26:45 vilhu001> */
/*****
/* Do QA for the LDB read in */
*****/

/* Dates are hard coded from 1990 to 2001 on purpose */
/* DO NOT CHANGE */

%macro set_ldb_qa;
```

Check to see if ANY of the files exist

```

%let numldb=0;
%do year=1990 %to 2001;
  %do q=1 %to 4;
%let filename=INPUTS.ldb_&state._&year.q&q.;
%let dsid=%sysfunc(open(&filename.,i));
%if (&dsid ^= 0) %then %do;
  %let numldb=%eval(&numldb+1);
  %let ldbobs=%sysfunc(attrn(&dsid.,nobs));

  %if &ldbobs>0 %then %do;
    data tmp;
      length message $ 50 ldbobs 8. yearqtr $6.;
message="&filename. &ldbobs obs used";
      ldbobs=&ldbobs;
      yearqtr="&year.q&q";
run;
      proc append base=ldbqabase data=tmp;
run;
%end;
%end;

%end; /* end q loop */
%end; /* end year loop */

/* If no LDB exists then create a dummy file */
%if (&numldb=0) %then %do;
%put LEHD WARNING ;
%put LEHD WARNING ;
%put LEHD WARNING ;
%put LEHD WARNING NO LDB DATA FOUND !!!;
%put LEHD WARNING ;
%put LEHD WARNING ;
%put LEHD WARNING ;
data tmp;
  length message $ 50 ldbobs 8. yearqtr $6.;
message="NO LDB DATA FOUND";
run;
  proc append base=ldbqabase data=tmp;
run;

```

Close QA file

```

%end; /* end if not condition */

data INTERWRK.ecfqa_ldb_not_used;
  set ldbqabase;
run;

%mend;

```

1.5.3.49 library/macros/set_ldb.sas

```

/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >---+ */
/* ---< Location: /programs/production/dev1/current/ecf >---+ */
/* ---< File: library/macros/set_ldb.sas >---+ */
/* Time-stamp: <04/09/17 17:02:29 vilhu001> */
/*****
/* Create the set statement for the LDB read in */
*****/

/* Dates are hard coded from 1990 to 2001 on purpose */
/* DO NOT CHANGE */

%macro set_ldb;

```

Check to see if ANY of the files exist

```

%let numldb=0;
%do year=1990 %to 2001;
  %do q=1 %to 4;
%let filename=INPUTS.ldb_&state._&year.q&q.;
%let dsid=%sysfunc(open(&filename.,i));
%if (&dsid ^= 0) %then %do;
  %let numldb=%eval(&numldb+1);
%end;
  %end; /* end q loop */
%end; /* end year loop */

```

If at least one LDB file exists then process

```

%if (&numldb>0) %then %do;
  /* Create the set statement */
  set
    %do year=1990 %to 2001;
      %do q=1 %to 4;
        %let filename=INPUTS.ldb_&state._&year.q&q.;
%let dsid=%sysfunc(open(&filename.,i));
%if (&dsid ^= 0) %then %do;
      &filename. (keep=sein year quarter seinunit naics
                  rename=(naics=naics_ldb))
        %end; /* end dsid condition */
      %end; /* end q loop */
    %end; /* end year loop */
  ;
%end; /* end numdb exist condition */

/* If no LDB exists then create a dummy file */
%if (&numldb=0) %then %do;
%put LEHD WARNING ;
%put LEHD WARNING ;
%put LEHD WARNING ;
%put LEHD WARNING NO LDB DATA FOUND !!!;
%put LEHD WARNING ;
%put LEHD WARNING ;
%put LEHD WARNING ;
  length sein $12 seinunit $5 year 4 quarter 3 naics_ldb $6;
  sein=" ";
  seinunit=" ";
  year=.;
  quarter=.;
  naics_ldb="999999";
%end; /* end if not condition */

```

Close QA file

```
%mend;
```

1.5.3.50 library/macros/state_specific_cleanup.sas

```
/* ---< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--- */
/* ---< Location: /programs/production/dev1/current/ecf >--- */
/* ---< File: library/macros/state_specific_cleanup.sas >--- */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/*****
* State specific code to clean up county, ein, naics, and sic
* If einavail= then setup dummy ein variable
```

```

Peter Welbrock changed the following code on 5/13/03 to incorporate
the standard of surrounding state based macro variable calls with
double quotes. This does (and did) cause problems with OR (Oregon)
*****/
```

```
%macro state_specific_cleanup;
/* State specific cleanup */
%if "&state"="ca" %then %do;
    if year=1900 then year=1995;
    if trim(county)="0" then county="ZZZ";
    if trim(county)="997" then county="ZZZ";
%end;
%if "&state"="fl" %then %do;
    *if year<2000 and county="086" then county="025";
    *else if year>=2000 and county="025" then county="086";
    if county="025" then county="086";
%end;
%if "&state"="va" %then %do;
    if county="560" then county="005";
%end;
%if "&state"="mn" %then %do;
    if trim(county)=" " then county="ZZZ";
    if trim(county)="619" then county="ZZZ";
%end;
%if "&state"="nc" %then %do;
    if trim(naics)="0" then naics="999999";
%end;
%if "&state"="tx" %then %do;
    if sic="9900" then sic="9999";
%end;
%if &einavail~=ein %then %do;
    length ein $9;
    ein=" ";
%end;
%mend;
```

1.5.3.51 library/macros/state_specific_keeprec.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/state_specific_keeprec.sas >--+ */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/*****
 * State specific code to remove duplicate records
 * Also include code to get rid of SEIN_BAD records
 *****/

%macro keeprec;
  %if "&state"="ca" %then %do;
    if first.seinunit then output;
  %end;
  %if "&state"="fl" %then %do;
    if first.seinunit then output;
  %end;
  %if "&state"="il" %then %do;
    if not(sein_bad=1 or (duprec=1 and SIC="9999")) then output;
  %end;
  %if "&state"="mn" %then %do;
    if sein_bad=0 or first.seinunit then output;
  %end;
%mend;
```

1.5.3.52 library/macros/upnum.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/upnum.sas >--+ */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/*****
*** MACRO to increment the transaction number ****
*****/

%macro upnum(dataset,vname);
  *** Use Proc Contents to get the number of observations ***;
  proc contents data=&dataset out=contents1;
  run;

  data _NULL_;
    set contents1;
    if trim(left(name))="&vname" then call symput('fuzz_obs',put(nobs,8.));
  run;

  *** If the dataset is empty then update transaction=0 ***;
  *** If the dataset is NOT empty then add 1 to the current ***;
  *** update value. ****;

  %if &fuzz_obs>0 %then %do;
    proc means data=&dataset;
      output out=upnum1 max(&vname)=old_max;
    run;

    proc print data=upnum1;
    run;

    data _NULL_;
      set upnum1;
      call symput('cupdte',put(old_max,8.));
    run;
    %let cupdte=%eval(&cupdte+1);
  %end;
  %else %do;
    %let cupdte=0;
  %end;
  %put Current Update Number is: &cupdte;
%mend;
```

1.5.3.53 library/macros/yqreal.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/yqreal.sas >--+ */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/*****
/** MACROS to convert qtime to year,quarter and back ****/
*****/

%macro yqreal(qtime);
    year=ceil(&qtime/4)+&ymin-1;
    quarter=&qtime-((year-&ymin)*4);
%mend;
```

1.5.3.54 library/macros/yrqtr.sas

```
/* +--< LEHD-QWI ecf 3.1.49 2005-01-26 schwa305 >--+ */
/* +--< Location: /programs/production/dev1/current/ecf >--+ */
/* +--< File: library/macros/yrqtr.sas >--+ */
/* Time-stamp: <03/11/02 10:26:50 vilhuber> */
/* FUNCTION TO CALCULATE YEAR,QUARTER for a given QTIME ***/
%macro yrqtr(qtime);
    yrtmp=ceil(&qtime/4)+&ymin-1;
    qrtmp=&qtime-((yrtmp-&ymin)*4);
%mend;
```

1.5.4 SAS layouts used

1.5.5 SAS formats used

1.5.6 Scripts used

1.5.7 Total code lines for this process

SAS	:	11055
KSH	:	0